Data manipulation with pipes

Luna L Sanchez Reyes

2023-03-07

The classic way of running code

For example I want the square root of the mean of a sequence of numbers

Nested code

```
numbers <- 1:300
mean(numbers)
```

```
## [1] 150.5
```

```
sqrt(mean(numbers))
```

```
## [1] 12.26784
```

Sequential code

In this case we create intermediate variables

```
numbers <- -300:546
numbers <- 1:300
numbers_mean <- mean(numbers)
sqrt(x = numbers_mean)
```

```
## [1] 12.26784
```

Piping code

It can be implemented in R using the package `magrittr`. It is a dependency of `dplyr`, so it is installed along.

```
library(magrittr)
```

The original symbol of the pipe is `%>%`. But we also have a new symbol that is similar to bash `|>` The purpose of pipes is to eliminate or reduce to the max the need of intermediate variables. For the mean example

```
1:300 %>% mean() %>% sqrt()
```

```
## [1] 12.26784
```

Pipes with the surveys data set

```
surveys <- read.csv(file = "../data-raw/surveys.csv")
str(surveys)
```

```
## 'data.frame':    35549 obs. of  9 variables:
##  $ record_id      : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ month          : int  7 7 7 7 7 7 7 7 7 7 ...
##  $ day            : int  16 16 16 16 16 16 16 16 16 16 ...
##  $ year           : int  1977 1977 1977 1977 1977 1977 1977 1977 1977 1977 ...
##  $ plot_id        : int  2 3 2 7 3 1 2 1 1 6 ...
##  $ species_id     : chr  "NL" "NL" "DM" "DM" ...
##  $ sex            : chr  "M" "M" "F" "M" ...
##  $ hindfoot_length: int  32 33 37 36 35 14 NA 37 34 20 ...
##  $ weight         : int  NA NA NA NA NA NA NA NA NA NA ...
```

Calculate the mean of the year column using pipes

```
surveys$year %>% mean()
```

```
## [1] 1990.475
```

Calculate the mean of the weight column:

```
surveys$weight %>% mean(na.rm = TRUE)
```

```
## [1] 42.67243
```

```
# ?mean
```

Exercise 1

1. Load surveys.csv into R using read.csv().

```
surveys <- read.csv(file = "../data-raw/surveys.csv")
```

2. Use select() to create a new data frame object called surveys1 with just the year, month, day, and species_id columns in that order.

```
surveys1 <- select(surveys, year, month, day, species_id)
str(surveys1)
```

```
## 'data.frame':    35549 obs. of  4 variables:
##  $ year      : int  1977 1977 1977 1977 1977 1977 1977 1977 1977 1977 ...
##  $ month     : int  7 7 7 7 7 7 7 7 7 7 ...
##  $ day       : int  16 16 16 16 16 16 16 16 16 16 ...
##  $ species_id: chr  "NL" "NL" "DM" "DM" ...
```

3. Create a new data frame called surveys2 with the year, species_id, and weight in kilograms of each individual, with no null weights. Use mutate(), select(), and filter() with !is.na(). The weight in the table is given in grams so you will need to create a new column called "weight_kg" for weight in kilograms by dividing the weight column by 1000.

```
surveys2 <- select(surveys, year, species_id, weight)
str(surveys2)
```

```
## 'data.frame':    35549 obs. of  3 variables:
##  $ year      : int  1977 1977 1977 1977 1977 1977 1977 1977 1977 1977 ...
##  $ species_id: chr  "NL" "NL" "DM" "DM" ...
##  $ weight    : int  NA NA NA NA NA NA NA NA NA NA ...
```

```
surveys2 <- mutate(surveys2, weight_kg = weight/1000)
str(surveys2)
```

```
## 'data.frame':    35549 obs. of  4 variables:
##  $ year      : int  1977 1977 1977 1977 1977 1977 1977 1977 1977 1977 ...
##  $ species_id: chr  "NL" "NL" "DM" "DM" ...
##  $ weight    : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ weight_kg : num  NA NA NA NA NA NA NA NA NA NA ...
```

```
surveys2 <- filter(surveys2, !is.na(weight_kg))
str(surveys2)
```

```
## 'data.frame':    32283 obs. of  4 variables:
##  $ year      : int  1977 1977 1977 1977 1977 1977 1977 1977 1977 1977 ...
##  $ species_id: chr  "DM" "DM" "DM" "DM" ...
##  $ weight    : int  40 48 29 46 36 52 8 22 35 7 ...
##  $ weight_kg : num  0.04 0.048 0.029 0.046 0.036 0.052 0.008 0.022 0.035 0.007 ...
```

```
surveys2 <- select(surveys2, year, species_id, weight_kg)
colnames(surveys2)
```

```
## [1] "year"       "species_id" "weight_kg"
```

```
# surveys2[ , c(1,3)]
# surveys2[ , c("year", "weight_kg")]
```

4. Use the filter() function to get all of the rows in the data frame surveys2 for the species ID “SH”.

```
surveys2_filtered <- filter(surveys2, species_id == "SH")
str(surveys2_filtered)
```

```
## 'data.frame':    141 obs. of  3 variables:
##  $ year      : int  1978 1982 1982 1986 1987 1987 1987 1987 1987 1988 ...
##  $ species_id: chr  "SH" "SH" "SH" "SH" ...
##  $ weight_kg : num  0.089 0.106 0.052 0.055 0.077 0.078 0.104 0.058 0.052 0.06 ...
```

Exercise 2

Redo all of the above but using pipes

```
read.csv(file = "../data-raw/surveys.csv") %>% # Read the data set
  select(year, month, day, species_id) -> surveys1 # select columns and assign to object

# name_object <- code that we want to run
str(surveys1)
```

```
## 'data.frame':    35549 obs. of  4 variables:
##  $ year      : int  1977 1977 1977 1977 1977 1977 1977 1977 1977 1977 ...
##  $ month     : int  7 7 7 7 7 7 7 7 7 7 ...
##  $ day       : int  16 16 16 16 16 16 16 16 16 16 ...
##  $ species_id: chr  "NL" "NL" "DM" "DM" ...
```

```
surveys %>% select(year, species_id, weight) %>%
  mutate(weight_kg = weight/1000) %>%
  filter(!is.na(weight)) %>%
  select(year, species_id, weight_kg) %>%
  filter(species_id == "SH") -> surveys_final

str(surveys_final)
```

```
## 'data.frame':    141 obs. of  3 variables:
##  $ year      : int  1978 1982 1982 1986 1987 1987 1987 1987 1987 1988 ...
##  $ species_id: chr  "SH" "SH" "SH" "SH" ...
##  $ weight_kg : num  0.089 0.106 0.052 0.055 0.077 0.078 0.104 0.058 0.052 0.06 ...
```

Exercise 3

Reformat the following code in pipe mode:

```
ds_data <- filter(surveys, species_id == "DS", !is.na(weight))
ds_data_by_year <- arrange(ds_data, year)
ds_weight_by_year <- select(ds_data_by_year, year, weight)
```

```
read.csv(file = "../data-raw/surveys.csv") %>%
  filter(species_id == "DS", !is.na(weight)) %>%
  arrange(year) %>%
  select(year, weight) -> final_table
str(final_table)
```

```
## 'data.frame':    2344 obs. of  2 variables:
##  $ year  : int  1977 1977 1977 1977 1977 1977 1977 1977 1977 1977 ...
##  $ weight: int  117 121 115 120 118 126 132 113 122 107 ...
```

```
head(final_table)
```

```
##   year weight
## 1 1977    117
## 2 1977    121
## 3 1977    115
## 4 1977    120
## 5 1977    118
## 6 1977    126
```

Piping to an argument that is not the first one

Some functions do not take the data as the first argument. For example the `lm()` function

```
str(surveys)
```

```
## 'data.frame':    35549 obs. of  9 variables:
##  $ record_id      : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ month          : int  7 7 7 7 7 7 7 7 7 7 ...
##  $ day            : int  16 16 16 16 16 16 16 16 16 16 ...
##  $ year           : int  1977 1977 1977 1977 1977 1977 1977 1977 1977 1977 ...
##  $ plot_id        : int  2 3 2 7 3 1 2 1 1 6 ...
##  $ species_id     : chr  "NL" "NL" "DM" "DM" ...
##  $ sex            : chr  "M" "M" "F" "M" ...
##  $ hindfoot_length: int  32 33 37 36 35 14 NA 37 34 20 ...
##  $ weight         : int  NA NA NA NA NA NA NA NA NA NA ...
```

```
lm(formula = weight ~ year, data = surveys)
```

```
##
## Call:
## lm(formula = weight ~ year, data = surveys)
##
## Coefficients:
## (Intercept)         year
##    2752.137       -1.361
```

And with a pipe:

```
surveys %>%
  lm(formula = weight ~ year, data = .) # use an underscore if you are using this pipe |>
```

```
##
## Call:
## lm(formula = weight ~ year, data = .)
##
## Coefficients:
## (Intercept)         year
##    2752.137       -1.361
```

Shortcut to create an r block in mac is `Ctrl + Option + i` In Windows is `Ctrl + Alt + i`

Exercise 4

```
surveys %>%
  filter(species_id == "DS", !is.na(weight)) %>%
  lm(weight ~ year, data = .) %>%
  summary()
```

```
##
## Call:
## lm(formula = weight ~ year, data = .)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -109.787  -12.440    3.723   14.886   69.886
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -709.1968   263.2510  -2.694  0.00711 **
## year           0.4184     0.1328   3.150  0.00165 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 22.86 on 2342 degrees of freedom
## Multiple R-squared:  0.00422,    Adjusted R-squared:  0.003795
## F-statistic: 9.925 on 1 and 2342 DF,  p-value: 0.001651
```

Grouping data or data aggregation

To get summary statistics for variables withing certain groups, we can group our dta by a certain value. For that we use the function `group_by()`

```
str(surveys)
```

```
## 'data.frame':    35549 obs. of  9 variables:
##  $ record_id      : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ month          : int  7 7 7 7 7 7 7 7 7 7 ...
##  $ day            : int  16 16 16 16 16 16 16 16 16 16 ...
##  $ year           : int  1977 1977 1977 1977 1977 1977 1977 1977 1977 1977 ...
##  $ plot_id        : int  2 3 2 7 3 1 2 1 1 6 ...
##  $ species_id     : chr  "NL" "NL" "DM" "DM" ...
##  $ sex            : chr  "M" "M" "F" "M" ...
##  $ hindfoot_length: int  32 33 37 36 35 14 NA 37 34 20 ...
##  $ weight         : int  NA NA NA NA NA NA NA NA NA NA ...
```

```
group_by(.data = surveys, year)
```

```
## # A tibble: 35,549 x 9
## # Groups:   year [26]
##    record_id month   day  year plot_id species_id sex   hindfoot_length weight
##        <int> <int> <int> <int>   <int> <chr>      <chr>           <int>  <int>
##  1         1     7    16  1977       2 NL         M                  32     NA
##  2         2     7    16  1977       3 NL         M                  33     NA
##  3         3     7    16  1977       2 DM         F                  37     NA
##  4         4     7    16  1977       7 DM         M                  36     NA
##  5         5     7    16  1977       3 DM         M                  35     NA
##  6         6     7    16  1977       1 PF         M                  14     NA
##  7         7     7    16  1977       2 PE         F                  NA     NA
##  8         8     7    16  1977       1 DM         M                  37     NA
##  9         9     7    16  1977       1 DM         F                  34     NA
## 10        10     7    16  1977       6 PF         F                  20     NA
## # ... with 35,539 more rows
```

```
grouped_surveys <- group_by(surveys, year)
str(grouped_surveys)
```

```
## gropd_df [35,549 x 9] (S3: grouped_df/tbl_df/tbl/data.frame)
##  $ record_id      : int [1:35549] 1 2 3 4 5 6 7 8 9 10 ...
##  $ month          : int [1:35549] 7 7 7 7 7 7 7 7 7 7 ...
##  $ day            : int [1:35549] 16 16 16 16 16 16 16 16 16 16 ...
##  $ year           : int [1:35549] 1977 1977 1977 1977 1977 1977 1977 1977 1977 1977 ...
##  $ plot_id        : int [1:35549] 2 3 2 7 3 1 2 1 1 6 ...
##  $ species_id     : chr [1:35549] "NL" "NL" "DM" "DM" ...
##  $ sex            : chr [1:35549] "M" "M" "F" "M" ...
##  $ hindfoot_length: int [1:35549] 32 33 37 36 35 14 NA 37 34 20 ...
##  $ weight         : int [1:35549] NA NA NA NA NA NA NA NA NA NA ...
##  - attr(*, "groups")= tibble [26 x 2] (S3: tbl_df/tbl/data.frame)
##   ..$ year : int [1:26] 1977 1978 1979 1980 1981 1982 1983 1984 1985 1986 ...
##   ..$ .rows: list<int> [1:26]
##   .. ..$ : int [1:503] 1 2 3 4 5 6 7 8 9 10 ...
##   .. ..$ : int [1:1048] 504 505 506 507 508 509 510 511 512 513 ...
##   .. ..$ : int [1:719] 1552 1553 1554 1555 1556 1557 1558 1559 1560 1561 ...
##   .. ..$ : int [1:1415] 2271 2272 2273 2274 2275 2276 2277 2278 2279 2280 ...
##   .. ..$ : int [1:1472] 3686 3687 3688 3689 3690 3691 3692 3693 3694 3695 ...
##   .. ..$ : int [1:1978] 5158 5159 5160 5161 5162 5163 5164 5165 5166 5167 ...
##   .. ..$ : int [1:1673] 7136 7137 7138 7139 7140 7141 7142 7143 7144 7145 ...
##   .. ..$ : int [1:981] 8809 8810 8811 8812 8813 8814 8815 8816 8817 8818 ...
##   .. ..$ : int [1:1438] 9790 9791 9792 9793 9794 9795 9796 9797 9798 9799 ...
##   .. ..$ : int [1:942] 11228 11229 11230 11231 11232 11233 11234 11235 11236 11237 ...
##   .. ..$ : int [1:1671] 12170 12171 12172 12173 12174 12175 12176 12177 12178 12179 ...
##   .. ..$ : int [1:1469] 13841 13842 13843 13844 13845 13846 13847 13848 13849 13850 ...
##   .. ..$ : int [1:1569] 15310 15311 15312 15313 15314 15315 15316 15317 15318 15319 ...
##   .. ..$ : int [1:1311] 16879 16880 16881 16882 16883 16884 16885 16886 16887 16888 ...
##   .. ..$ : int [1:1347] 18190 18191 18192 18193 18194 18195 18196 18197 18198 18199 ...
##   .. ..$ : int [1:1038] 19537 19538 19539 19540 19541 19542 19543 19544 19545 19546 ...
##   .. ..$ : int [1:750] 20575 20576 20577 20578 20579 20580 20581 20582 20583 20584 ...
##   .. ..$ : int [1:668] 21325 21326 21327 21328 21329 21330 21331 21332 21333 21334 ...
##   .. ..$ : int [1:1222] 21993 21994 21995 21996 21997 21998 21999 22000 22001 22002 ...
##   .. ..$ : int [1:1706] 23215 23216 23217 23218 23219 23220 23221 23222 23223 23224 ...
##   .. ..$ : int [1:2493] 24921 24922 24923 24924 24925 24926 24927 24928 24929 24930 ...
##   .. ..$ : int [1:1610] 27414 27415 27416 27417 27418 27419 27420 27421 27422 27423 ...
##   .. ..$ : int [1:1135] 29024 29025 29026 29027 29028 29029 29030 29031 29032 29033 ...
##   .. ..$ : int [1:1552] 30159 30160 30161 30162 30163 30164 30165 30166 30167 30168 ...
##   .. ..$ : int [1:1610] 31711 31712 31713 31714 31715 31716 31717 31718 31719 31720 ...
##   .. ..$ : int [1:2229] 33321 33322 33323 33324 33325 33326 33327 33328 33329 33330 ...
##   .. ..@ ptype: int(0)
##   ..- attr(*, ".drop")= logi TRUE
```

```
group_by(surveys, year, sex)
```

```
## # A tibble: 35,549 x 9
## # Groups:   year, sex [78]
##    record_id month   day  year plot_id species_id sex   hindfoot_length weight
##        <int> <int> <int> <int>   <int> <chr>      <chr>           <int>  <int>
##  1         1     7    16  1977       2 NL         M                  32     NA
##  2         2     7    16  1977       3 NL         M                  33     NA
##  3         3     7    16  1977       2 DM         F                  37     NA
##  4         4     7    16  1977       7 DM         M                  36     NA
##  5         5     7    16  1977       3 DM         M                  35     NA
##  6         6     7    16  1977       1 PF         M                  14     NA
##  7         7     7    16  1977       2 PE         F                  NA     NA
##  8         8     7    16  1977       1 DM         M                  37     NA
##  9         9     7    16  1977       1 DM         F                  34     NA
## 10        10     7    16  1977       6 PF         F                  20     NA
## # ... with 35,539 more rows
```

Get summary statistics of groups

The summary function create a new table that has the summary statistyics that we asked for per each group on the tibble.

```
group_by(surveys, year, sex) %>%
  summarize(diversity = n())
```

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

```
## # A tibble: 78 x 3
## # Groups:   year [26]
##     year sex   diversity
##    <int> <chr>     <int>
##  1  1977 ""           85
##  2  1977 "F"         204
##  3  1977 "M"         214
##  4  1978 ""          112
##  5  1978 "F"         503
##  6  1978 "M"         433
##  7  1979 ""           68
##  8  1979 "F"         327
##  9  1979 "M"         324
## 10  1980 ""           83
## # ... with 68 more rows
```

```
group_by(surveys, year) %>%
  summarize(mean_weight = mean(weight, na.rm = TRUE))
```

```
## # A tibble: 26 x 2
##     year mean_weight
##    <int>       <dbl>
##  1  1977        46.7
##  2  1978        67.9
##  3  1979        63.4
##  4  1980        62.4
##  5  1981        65.8
##  6  1982        53.8
##  7  1983        55.1
##  8  1984        51.0
##  9  1985        46.7
## 10  1986        55.1
## # ... with 16 more rows
```

Exercise 5

1. Load surveys.csv into R using read.csv() and assign it to an object called surveys.
2. Use the group_by() and summarize() functions to get a count of the number of individuals with each species ID.

```
str(surveys)
```

```
## 'data.frame':    35549 obs. of  9 variables:
##  $ record_id      : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ month          : int  7 7 7 7 7 7 7 7 7 7 ...
##  $ day            : int  16 16 16 16 16 16 16 16 16 16 ...
##  $ year           : int  1977 1977 1977 1977 1977 1977 1977 1977 1977 1977 ...
##  $ plot_id        : int  2 3 2 7 3 1 2 1 1 6 ...
##  $ species_id     : chr  "NL" "NL" "DM" "DM" ...
##  $ sex            : chr  "M" "M" "F" "M" ...
##  $ hindfoot_length: int  32 33 37 36 35 14 NA 37 34 20 ...
##  $ weight         : int  NA NA NA NA NA NA NA NA NA NA ...
```

```
read.csv(file = "../data-raw/surveys.csv") %>%
  group_by(species_id) %>%
  summarize(count = n())
```

```
## # A tibble: 49 x 2
##    species_id count
##    <chr>      <int>
##  1 ""           763
##  2 "AB"         303
##  3 "AH"         437
##  4 "AS"           2
##  5 "BA"          46
##  6 "CB"          50
##  7 "CM"          13
##  8 "CQ"          16
##  9 "CS"           1
## 10 "CT"           1
## # ... with 39 more rows
```

3. Use the group_by() and summarize() functions to get a count of the number of individuals with each species ID in each year.

```
str(surveys)
```

```
## 'data.frame':    35549 obs. of  9 variables:
##  $ record_id      : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ month          : int  7 7 7 7 7 7 7 7 7 7 ...
##  $ day            : int  16 16 16 16 16 16 16 16 16 16 ...
##  $ year           : int  1977 1977 1977 1977 1977 1977 1977 1977 1977 1977 ...
##  $ plot_id        : int  2 3 2 7 3 1 2 1 1 6 ...
##  $ species_id     : chr  "NL" "NL" "DM" "DM" ...
##  $ sex            : chr  "M" "M" "F" "M" ...
##  $ hindfoot_length: int  32 33 37 36 35 14 NA 37 34 20 ...
##  $ weight         : int  NA NA NA NA NA NA NA NA NA NA ...
```

```r
read.csv(file = "../data-raw/surveys.csv") %>%
  group_by(species_id, year) %>%
  summarize(count = n())
```

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

```
## # A tibble: 535 x 3
## # Groups:   species_id [49]
##    species_id  year count
##    <chr>      <int> <int>
##  1 ""          1977    16
##  2 ""          1978    56
##  3 ""          1979    61
##  4 ""          1980    40
##  5 ""          1981    55
##  6 ""          1982    14
##  7 ""          1983    21
##  8 ""          1984    30
##  9 ""          1985    22
## 10 ""          1986    20
## # ... with 525 more rows
```

4. Use the filter(), group_by(), and summarize() functions to get the mean mass of species with species_id equals to DO in each year.

```r
my_result <- read.csv(file = "../data-raw/surveys.csv") %>%
  filter(species_id == "DO") %>%
  group_by(year) %>%
  summarize(mean_mass = mean(weight, na.rm = TRUE))

my_result
```

```
## # A tibble: 26 x 2
##     year mean_mass
##    <int>     <dbl>
##  1  1977      42.7
##  2  1978      45
##  3  1979      45.9
##  4  1980      48.1
##  5  1981      49.1
##  6  1982      47.9
##  7  1983      47.2
##  8  1984      48.4
##  9  1985      48.0
## 10  1986      49.4
## # ... with 16 more rows
```

```r
read.csv(file = "../data-raw/surveys.csv") %>%
  group_by(year) %>%
  filter(species_id == "DO") %>%
  summarize(mean_mass = mean(weight, na.rm = TRUE)) -> my_result
```