Tidying with James

JM Muriuki

2024-06-29

Here we'll learn some tools to help make our data tidy and more coder-friendly

1. Use 'r tidyr::pivot_wider()' and 'r tidy::pivot_longer()' to reshape data frames
2. 'r janitor::clean_names()' to make column headers more manageable
3. 'r tidyr::unite()' and 'r tidyr::separate()' to merge or separate information from different columns
4. Detect or replace a string with 'r stringr' functions

Attach packages

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(here)
```

```
## here() starts at C:/Users/admin/OneDrive/Documents/Data Analyst James/R for Data Science
```

```
library(janitor)
```

```
## Warning: package 'janitor' was built under R version 4.3.3
```

```
##
## Attaching package: 'janitor'
##
## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test
```

```
library(readxl)
```

read_excel() to read in data from an Excel worksheet

```
inverts <- read_excel(here("inverts.xlsx"))
```

explore the imported data

```
view(inverts)
names(inverts)
```

```
## [1] "month"       "site"        "common_name" "2016"        "2017"
## [6] "2018"
```

```
summary(inverts)
```

```
##     month               site           common_name            2016
##  Length:55          Length:55          Length:55          Min.   :  16.0
##  Class :character   Class :character   Class :character   1st Qu.:  24.0
##  Mode  :character   Mode  :character   Mode  :character   Median : 255.0
##                                                           Mean   : 697.8
##                                                           3rd Qu.: 815.0
##                                                           Max.   :6384.0
##
##       2017             2018
##  Min.   :  16.0   Min.   :  16.0
##  1st Qu.:  24.0   1st Qu.:  24.0
##  Median :  73.0   Median : 242.5
##  Mean   : 658.1   Mean   : 981.3
##  3rd Qu.:1010.5   3rd Qu.:1389.8
##  Max.   :4398.0   Max.   :4955.0
##                   NA's   :5
```

tidyr::pivot_longer() to reshape from wider-to-longer format

```
# we'll use tidyr::pivot_longer() to gather data from all years in inverts(columns 2016,2017,and 2018)
```

1. One called year, which contains the year
2. One called sp_count containing the number of each species observed

```
# Note: Either single-quotes, double-quoted , or backticks around years work!
inverts_long <- pivot_longer(data=inverts,cols='2016':'2018',names_to = "year", values_to = "sp_count")
inverts_long
```

```
## # A tibble: 165 x 5
##    month site  common_name              year  sp_count
##    <chr> <chr> <chr>                    <chr>    <dbl>
##  1 7     abur  california cone snail    2016       451
##  2 7     abur  california cone snail    2017        28
##  3 7     abur  california cone snail    2018       762
##  4 7     abur  california spiny lobster 2016        17
##  5 7     abur  california spiny lobster 2017        17
##  6 7     abur  california spiny lobster 2018        16
##  7 7     abur  orange cup coral         2016        24
##  8 7     abur  orange cup coral         2017        24
##  9 7     abur  orange cup coral         2018        24
## 10 7     abur  purple urchin            2016        48
## # i 155 more rows
```

Explore the class of year in inverts_long:

```
class(inverts_long$year)
```

```
## [1] "character"
```

we'll use mutate() to add a column called year, which contains an 'r as.numeric()' version of the existing year variable

```
# Coerce "year" class to numeric:
inverts_long <- inverts_long %>% mutate(year=as.numeric(year))

# checking the class again, we see that year has been updated to a numeric variable:
class(inverts_long$year)
```

```
## [1] "numeric"
```

tidyr::pivot_wider() to convert from longer-to-wider format

We want each species in the common_name column to exist as its own column. In that case, we would be converting from a longer to a wider format , and will use tidyr::pivot_wider()

```
inverts_wide <- inverts_long %>% pivot_wider(names_from = common_name,values_from = sp_count)
inverts_wide
```

```
## # A tibble: 33 x 8
##    month site   year `california cone snail` `california spiny lobster`
##    <chr> <chr> <dbl>                   <dbl>                       <dbl>
##  1 7     abur   2016                     451                          17
##  2 7     abur   2017                      28                          17
##  3 7     abur   2018                     762                          16
##  4 7     ahnd   2016                      27                          16
```

```
##  5 7     ahnd   2017                   24                          16
##  6 7     ahnd   2018                   24                          16
##  7 7     aque   2016                 4971                          48
##  8 7     aque   2017                 1752                          48
##  9 7     aque   2018                 2616                          48
## 10 7     bull   2016                 1735                          24
## # i 23 more rows
## # i 3 more variables: `orange cup coral` <dbl>, `purple urchin` <dbl>,
## #   `rock scallop` <dbl>
```

janitor::clean_names() to clean up column names

The 'r janitor' package by Sam Firke is a great collection of functions for some quick data cleaning like:

1. 'r janitor::clean_names()' : update column headers to a case of your choosing
2. 'r janitor::get_dupes()' : see all rows that are duplicates within variables you choose
3. 'r janitor::remove_empty()' : remove empty rows and/or columns
4. 'r janitor::adorn_*()' : jazz up tables

Here, we'll use janitor::clean_names() to convert all of our column headers to a more convenient case - the default is lower_snake_case, which means all spaces and symbols are replaced with an underscore (or a word describing the symbol), all characters are lowercase, and a few other nice adjustments.

For example, janitor::clean_names() would update these nightmare column names into much nicer forms:

```
My...RECENT-income! becomes my_recent_income
SAMPLE2.!test1 becomes sample2_test1
ThisIsTheName becomes this_is_the_name
2015 becomes x2015
```

```r
inverts_wide <- inverts_wide %>% clean_names()
names(inverts_wide)
```

```
## [1] "month"                   "site"
## [3] "year"                    "california_cone_snail"
## [5] "california_spiny_lobster" "orange_cup_coral"
## [7] "purple_urchin"           "rock_scallop"
```

And there are other case options in clean_names(), like:

```
"snake" produces snake_case (the default)
"lower_camel" or "small_camel" produces lowerCamel
"upper_camel" or "big_camel" produces UpperCamel
"screaming_snake" or "all_caps" produces ALL_CAPS
"lower_upper" produces lowerUPPER
"upper_lower" produces UPPERlower
```

tidyr::unite() and tidyr::separate() to combine or separate information in column(s)

For example, the following data frame has *genus* and *species* in separate columns: We may want to combine the genus and species into a single column , scientific_name:

Or we may want to do the reverse(separate information from a single column into multiple columns).Here, we'll learn tidyr::unite() and tidyr::separate() and tidyr::separate() to help us do both

tidyr::unite() to merge information from separate columns

```
inverts_unite <- inverts_long %>%  unite(col="site_year",# what to name the new united colum
                                         c(site,year), # The columns we'll unite(site,year)
                                         sep="-")# How to separate the things we're uniting
inverts_unite
```

```
## # A tibble: 165 x 4
##    month site_year common_name              sp_count
##    <chr> <chr>     <chr>                       <dbl>
##  1 7     abur-2016 california cone snail         451
##  2 7     abur-2017 california cone snail          28
##  3 7     abur-2018 california cone snail         762
##  4 7     abur-2016 california spiny lobster       17
##  5 7     abur-2017 california spiny lobster       17
##  6 7     abur-2018 california spiny lobster       16
##  7 7     abur-2016 orange cup coral               24
##  8 7     abur-2017 orange cup coral               24
##  9 7     abur-2018 orange cup coral               24
## 10 7     abur-2016 purple urchin                  48
## # i 155 more rows
```

Activity

Creating a new object called 'inverts_moyr', starting from inverts_long, that unites the month and year columns into a single column named "mo_yr," using a slash "/" as the separator. Then try updating the separator to something else! Like "hello!"

```
inverts_moyr <- inverts_long %>% unite(col = "mo_yr", # What to name the new united column
                                       c(month,year),# The columns we'll unite (site,year)
                                       sep="/")
inverts_moyr
```

```
## # A tibble: 165 x 4
##    mo_yr  site  common_name              sp_count
##    <chr>  <chr> <chr>                       <dbl>
##  1 7/2016 abur  california cone snail         451
##  2 7/2017 abur  california cone snail          28
##  3 7/2018 abur  california cone snail         762
##  4 7/2016 abur  california spiny lobster       17
##  5 7/2017 abur  california spiny lobster       17
##  6 7/2018 abur  california spiny lobster       16
```

```
##  7 7/2016 abur  orange cup coral               24
##  8 7/2017 abur  orange cup coral               24
##  9 7/2018 abur  orange cup coral               24
## 10 7/2016 abur  purple urchin                  48
## # i 155 more rows
```

Merging information from >2 columns (not done in workshop)

```
# Uniting more than 2 columns:
inverts_triple_unite <- inverts_long %>% unite(col = "year_site_name", c(year,site,common_name),sep="-")
head(inverts_triple_unite)
```

```
## # A tibble: 6 x 3
##   month year_site_name                      sp_count
##   <chr> <chr>                                  <dbl>
## 1 7     2016-abur-california cone snail          451
## 2 7     2017-abur-california cone snail           28
## 3 7     2018-abur-california cone snail          762
## 4 7     2016-abur-california spiny lobster        17
## 5 7     2017-abur-california spiny lobster        17
## 6 7     2018-abur-california spiny lobster        16
```

tidyr::separate() to separate information into multiple columns

```
inverts_sep <- inverts_unite %>% separate(site_year, into = c("my_site","my_year"))
head(inverts_sep)
```

```
## # A tibble: 6 x 5
##   month my_site my_year common_name              sp_count
##   <chr> <chr>   <chr>   <chr>                       <dbl>
## 1 7     abur    2016    california cone snail         451
## 2 7     abur    2017    california cone snail          28
## 3 7     abur    2018    california cone snail         762
## 4 7     abur    2016    california spiny lobster       17
## 5 7     abur    2017    california spiny lobster       17
## 6 7     abur    2018    california spiny lobster       16
```

stringr::str_replace() to replace a pattern

Did someone wrongly enter "fish" as "fsh" throughout the spreadsheet, and you want to update it everywhere?

Use 'r string::str_replace()' to automatically replace a string pattern

```
ca_abbr <- inverts %>% mutate(common_name=str_replace(common_name, pattern = "california", replacement
```