Sheet III: Case Study - Tidy Data

Dr. Falkenberg

WS 2019/20

Case Study: compare chapter 12.6 R for Data Science (Grolemund, Wickham)

1. load tidyverse and the data set who

```
library(tidyverse)
tidyr::who
```

```
## # A tibble: 7,240 x 60
##    country iso2  iso3   year new_sp_m014 new_sp_m1524 new_sp_m2534
##    <chr>   <chr> <chr> <int>       <int>        <int>        <int>
##  1 Afghan~ AF    AFG    1980          NA           NA           NA
##  2 Afghan~ AF    AFG    1981          NA           NA           NA
##  3 Afghan~ AF    AFG    1982          NA           NA           NA
##  4 Afghan~ AF    AFG    1983          NA           NA           NA
##  5 Afghan~ AF    AFG    1984          NA           NA           NA
##  6 Afghan~ AF    AFG    1985          NA           NA           NA
##  7 Afghan~ AF    AFG    1986          NA           NA           NA
##  8 Afghan~ AF    AFG    1987          NA           NA           NA
##  9 Afghan~ AF    AFG    1988          NA           NA           NA
## 10 Afghan~ AF    AFG    1989          NA           NA           NA
## # ... with 7,230 more rows, and 53 more variables: new_sp_m3544 <int>,
## #   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>, new_sn_f3544 <int>,
## #   new_sn_f4554 <int>, new_sn_f5564 <int>, new_sn_f65 <int>,
## #   new_ep_m014 <int>, new_ep_m1524 <int>, new_ep_m2534 <int>,
## #   new_ep_m3544 <int>, new_ep_m4554 <int>, new_ep_m5564 <int>,
## #   new_ep_m65 <int>, new_ep_f014 <int>, new_ep_f1524 <int>,
## #   new_ep_f2534 <int>, new_ep_f3544 <int>, new_ep_f4554 <int>,
## #   new_ep_f5564 <int>, new_ep_f65 <int>, newrel_m014 <int>,
## #   newrel_m1524 <int>, newrel_m2534 <int>, newrel_m3544 <int>,
## #   newrel_m4554 <int>, newrel_m5564 <int>, newrel_m65 <int>,
## #   newrel_f014 <int>, newrel_f1524 <int>, newrel_f2534 <int>,
## #   newrel_f3544 <int>, newrel_f4554 <int>, newrel_f5564 <int>,
## #   newrel_f65 <int>
```

2. Clean the data set in several steps

a) Identify columns that are not variables.

```
head(who)
```

```
## # A tibble: 6 x 60
##   country iso2  iso3   year new_sp_m014 new_sp_m1524 new_sp_m2534
##   <chr>   <chr> <chr> <int>       <int>        <int>        <int>
## 1 Afghan~ AF    AFG    1980          NA           NA           NA
## 2 Afghan~ AF    AFG    1981          NA           NA           NA
## 3 Afghan~ AF    AFG    1982          NA           NA           NA
## 4 Afghan~ AF    AFG    1983          NA           NA           NA
## 5 Afghan~ AF    AFG    1984          NA           NA           NA
## 6 Afghan~ AF    AFG    1985          NA           NA           NA
## # ... with 53 more variables: new_sp_m3544 <int>, 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>, new_sn_f3544 <int>, new_sn_f4554 <int>,
## #   new_sn_f5564 <int>, new_sn_f65 <int>, new_ep_m014 <int>,
## #   new_ep_m1524 <int>, new_ep_m2534 <int>, new_ep_m3544 <int>,
## #   new_ep_m4554 <int>, new_ep_m5564 <int>, new_ep_m65 <int>,
## #   new_ep_f014 <int>, new_ep_f1524 <int>, new_ep_f2534 <int>,
## #   new_ep_f3544 <int>, new_ep_f4554 <int>, new_ep_f5564 <int>,
## #   new_ep_f65 <int>, newrel_m014 <int>, newrel_m1524 <int>,
## #   newrel_m2534 <int>, newrel_m3544 <int>, newrel_m4554 <int>,
## #   newrel_m5564 <int>, newrel_m65 <int>, newrel_f014 <int>,
## #   newrel_f1524 <int>, newrel_f2534 <int>, newrel_f3544 <int>,
## #   newrel_f4554 <int>, newrel_f5564 <int>, newrel_f65 <int>
```

- Inspect the columns

```
?tidyr::who
```

```
## starting httpd help server ... done
```

World Health Organization TB data Description

A subset of data from the World Health Organization Global Tuberculosis Report, and accompanying global populations.

A dataset with the variables

- country: Country name
- iso2, iso3: 2 & 3 letter ISO country codes
- year: Year
- new_sp_m014 - new_rel_f65: Counts of new TB cases recorded by group. Column names encode three variables that describe the group (see details).

Details: The data uses the original codes given by the World Health Organization. The column names for columns five through 60 are made by combining new_ to a code for method of diagnosis (rel = relapse, sn = negative pulmonary smear, sp = positive pulmonary smear, ep = extrapulmonary) to a code for gender (f = female, m = male) to a code for age group (014 =

0-14 yrs of age, 1524 = 15-24 years of age, 2534 = 25 to 34 years of age, 3544 = 35 to 44 years of age, 4554 = 45 to 54 years of age, 5564 = 55 to 64 years of age, 65 = 65 years of age or older).

Which are columns are variables?

- country, iso2, and iso3 are three variables that redundantly specify the country.
- year is clearly also a variable.
- From the structure in the variable names (e.g. new_sp_m014, new_ep_m014, new_ep_f014, …) these are likely to be values, not variables.

b) Inspect the gather() command and apply the command to gather together all the columns from new_sp_m014 to newrel_f65. Since we do not know what the values represent, give them the generic name ``key’’. The cells represent the count of cases, therefore use the variable cases. Remove the missing values in the current representation using na.rm.

The gather() command

```
?gather()
```

gather(data, key = “key”, value = “value”, …, na.rm = FALSE, convert = FALSE, factor_key = FALSE)

Arguments

- data: A data frame.
- key, value: Names of new key and value columns, as strings or symbols.
- … : A selection of columns. If empty, all variables are selected. You can supply bare variable names, select all variables between x and z with x:z, exclude y with -y.
- na.rm: If TRUE, will remove rows from output where the value column in NA.
- convert: If TRUE will automatically run type.convert() on the key column. This is useful of the column names are actually numeric, integer, or logical.
- factor_key: If FALSE, the default, the key values will be stored as a character vector. If TRUE, will be stored as a factor, which preserves the original ordering of the columns.

apply gather()

```
who1 <- who %>%
  gather(key = "key", value = "cases",
         new_sp_m014:newrel_f65,
         na.rm = TRUE)
who1
```

```
## # A tibble: 76,046 x 6
##    country     iso2  iso3   year key          cases
##  * <chr>       <chr> <chr> <int> <chr>        <int>
##  1 Afghanistan AF    AFG    1997 new_sp_m014      0
##  2 Afghanistan AF    AFG    1998 new_sp_m014     30
##  3 Afghanistan AF    AFG    1999 new_sp_m014      8
##  4 Afghanistan AF    AFG    2000 new_sp_m014     52
##  5 Afghanistan AF    AFG    2001 new_sp_m014    129
##  6 Afghanistan AF    AFG    2002 new_sp_m014     90
##  7 Afghanistan AF    AFG    2003 new_sp_m014    127
##  8 Afghanistan AF    AFG    2004 new_sp_m014    139
##  9 Afghanistan AF    AFG    2005 new_sp_m014    151
## 10 Afghanistan AF    AFG    2006 new_sp_m014    193
## # ... with 76,036 more rows
```

c) Count the values in the new ``key'' column.

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

```
## # A tibble: 56 x 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
##  7 new_ep_f65    1014
##  8 new_ep_m014   1038
##  9 new_ep_m1524  1026
## 10 new_ep_m2534  1020
## # ... with 46 more rows
```

d) The values of the new column ``key'' have the following structure

The first three letters of each column denote whether the column contains new or old cases of TB. In this dataset, each column contains new cases.

The next two letters describe the type of TB:

- rel stands for cases of relapse
- ep stands for cases of extrapulmonary TB
- sn stands for cases of pulmonary TB that could not be diagnosed by a pulmonary smear (smear negative)
- sp stands for cases of pulmonary TB that could be diagnosed be a pulmonary smear (smear positive)

The sixth letter gives the sex of TB patients. The dataset groups cases by males (m) and females (f).

The remaining numbers gives the age group. The dataset groups cases into seven age groups:

- 014 = 0 - 14 years old
- 1524 = 15 - 24 years old
- 2534 = 25 - 34 years old

- 3544 = 35 - 44 years old
- 4554 = 45 - 54 years old
- 5564 = 55 - 64 years old
- 65 = 65 or older

Unfortunately the names are slightly inconsistent because instead of new_rel we have newrel. Use str_replace() command to replace the characters `newrel` with `new_rel`.

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

```
##  [1] "new_sp_m014"   "new_sp_m1524"  "new_sp_m2534"  "new_sp_m3544" 
##  [5] "new_sp_m4554"  "new_sp_m5564"  "new_sp_m65"    "new_sp_f014"  
##  [9] "new_sp_f1524"  "new_sp_f2534"  "new_sp_f3544"  "new_sp_f4554" 
## [13] "new_sp_f5564"  "new_sp_f65"    "new_sn_m014"   "new_sn_m1524" 
## [17] "new_sn_m2534"  "new_sn_m3544"  "new_sn_m4554"  "new_sn_m5564" 
## [21] "new_sn_m65"    "new_sn_f014"   "new_sn_f1524"  "new_sn_f2534" 
## [25] "new_sn_f3544"  "new_sn_f4554"  "new_sn_f5564"  "new_sn_f65"   
## [29] "new_ep_m014"   "new_ep_m1524"  "new_ep_m2534"  "new_ep_m3544" 
## [33] "new_ep_m4554"  "new_ep_m5564"  "new_ep_m65"    "new_ep_f014"  
## [37] "new_ep_f1524"  "new_ep_f2534"  "new_ep_f3544"  "new_ep_f4554" 
## [41] "new_ep_f5564"  "new_ep_f65"    "new_rel_m014"  "new_rel_m1524"
## [45] "new_rel_m2534" "new_rel_m3544" "new_rel_m4554" "new_rel_m5564"
## [49] "new_rel_m65"   "new_rel_f014"  "new_rel_f1524" "new_rel_f2534"
## [53] "new_rel_f3544" "new_rel_f4554" "new_rel_f5564" "new_rel_f65"
```

Split the codes at each underscore

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

```
## # A tibble: 76,046 x 8
##    country     iso2  iso3   year new   type  sexage cases
##    <chr>       <chr> <chr> <int> <chr> <chr> <chr>  <int>
##  1 Afghanistan AF    AFG    1997 new   sp    m014       0
##  2 Afghanistan AF    AFG    1998 new   sp    m014      30
##  3 Afghanistan AF    AFG    1999 new   sp    m014       8
##  4 Afghanistan AF    AFG    2000 new   sp    m014      52
##  5 Afghanistan AF    AFG    2001 new   sp    m014     129
##  6 Afghanistan AF    AFG    2002 new   sp    m014      90
##  7 Afghanistan AF    AFG    2003 new   sp    m014     127
##  8 Afghanistan AF    AFG    2004 new   sp    m014     139
##  9 Afghanistan AF    AFG    2005 new   sp    m014     151
## 10 Afghanistan AF    AFG    2006 new   sp    m014     193
## # ... with 76,036 more rows
```

Separate the values of sexage after the first character

```
who4 <- who3 %>%
  separate(sexage, c("sex", "age"), sep = 1)
who4
```

```
## # A tibble: 76,046 x 9
##    country     iso2  iso3   year new   type  sex   age   cases
##    <chr>       <chr> <chr> <int> <chr> <chr> <chr> <chr> <int>
##  1 Afghanistan AF    AFG    1997 new   sp    m     014       0
##  2 Afghanistan AF    AFG    1998 new   sp    m     014      30
##  3 Afghanistan AF    AFG    1999 new   sp    m     014       8
##  4 Afghanistan AF    AFG    2000 new   sp    m     014      52
##  5 Afghanistan AF    AFG    2001 new   sp    m     014     129
##  6 Afghanistan AF    AFG    2002 new   sp    m     014      90
##  7 Afghanistan AF    AFG    2003 new   sp    m     014     127
##  8 Afghanistan AF    AFG    2004 new   sp    m     014     139
##  9 Afghanistan AF    AFG    2005 new   sp    m     014     151
## 10 Afghanistan AF    AFG    2006 new   sp    m     014     193
## # ... with 76,036 more rows
```

e) Remove the redundant columns new, iso2 and iso3.

```
who5 <- who4 %>% select(-new, -iso2, -iso3)
who5
```

```
## # A tibble: 76,046 x 6
##    country      year type  sex   age   cases
##    <chr>       <int> <chr> <chr> <chr> <int>
##  1 Afghanistan  1997 sp    m     014       0
##  2 Afghanistan  1998 sp    m     014      30
##  3 Afghanistan  1999 sp    m     014       8
##  4 Afghanistan  2000 sp    m     014      52
##  5 Afghanistan  2001 sp    m     014     129
##  6 Afghanistan  2002 sp    m     014      90
##  7 Afghanistan  2003 sp    m     014     127
##  8 Afghanistan  2004 sp    m     014     139
##  9 Afghanistan  2005 sp    m     014     151
## 10 Afghanistan  2006 sp    m     014     193
## # ... with 76,036 more rows
```

3) Clean the data set using pipes

```
whocleaned <- who %>%
  gather(key = "key",
         value = "cases",
         new_sp_m014:newrel_f65, na.rm = TRUE) %>%
  mutate(key =
           stringr::str_replace(key,"newrel","new_rel")) %>%
  separate(key,c("new", "type", "sexage"), by = "_") %>%
  separate(sexage, c("sex", "age"), sep = 1) %>%
  select(-new, -iso2, -iso3)
whocleaned
```

```
## # A tibble: 76,046 x 6
##    country      year type  sex   age   cases
##    <chr>       <int> <chr> <chr> <chr> <int>
##  1 Afghanistan  1997 sp    m     014       0
##  2 Afghanistan  1998 sp    m     014      30
##  3 Afghanistan  1999 sp    m     014       8
##  4 Afghanistan  2000 sp    m     014      52
##  5 Afghanistan  2001 sp    m     014     129
##  6 Afghanistan  2002 sp    m     014      90
##  7 Afghanistan  2003 sp    m     014     127
##  8 Afghanistan  2004 sp    m     014     139
##  9 Afghanistan  2005 sp    m     014     151
## 10 Afghanistan  2006 sp    m     014     193
## # ... with 76,036 more rows
```

4) Create a table containing country and for every the population and the number of all infections. Use the function tally/count().

```
population
```

```
## # A tibble: 4,060 x 3
##    country      year population
##    <chr>       <int>      <int>
##  1 Afghanistan  1995   17586073
##  2 Afghanistan  1996   18415307
##  3 Afghanistan  1997   19021226
##  4 Afghanistan  1998   19496836
##  5 Afghanistan  1999   19987071
##  6 Afghanistan  2000   20595360
##  7 Afghanistan  2001   21347782
##  8 Afghanistan  2002   22202806
##  9 Afghanistan  2003   23116142
## 10 Afghanistan  2004   24018682
## # ... with 4,050 more rows
```

```
table1 <- right_join(
  population,
  whocleaned %>% count(country, year, wt = cases),
  by = c("country", "year")
)
table1
```

```
## # A tibble: 3,484 x 4
##    country      year population     n
##    <chr>       <int>      <int> <int>
##  1 Afghanistan  1997   19021226   128
##  2 Afghanistan  1998   19496836  1778
##  3 Afghanistan  1999   19987071   745
##  4 Afghanistan  2000   20595360  2666
##  5 Afghanistan  2001   21347782  4639
##  6 Afghanistan  2002   22202806  6509
##  7 Afghanistan  2003   23116142  6528
##  8 Afghanistan  2004   24018682  8245
##  9 Afghanistan  2005   24860855  9949
## 10 Afghanistan  2006   25631282 12469
## # ... with 3,474 more rows
```