Prof. Eric A. Suess

So how should you complete your homework for this class?

- First thing to do is type all of your information about the problems you do in the text part of your R Notebook.
- Second thing to do is type all of your R code into R chunks that can be run.
- If you load the tidyverse in an R Notebook chunk, be sure to include the “message = FALSE” in the \{r\}, so \{r message = FALSE\}.
- Last thing is to spell check your R Notebook. Edit > Check Spelling... or hit the F7 key.

Homework 8:

Read: Chapter 12

Do 12.2.1 Exercises 1, 2
Do 12.3.3 Exercise 4
Do 12.4.3 Exercise 1

```r
library(tidyverse)
```

### 12.2.1

1.

Using prose, describe how the variables and observations are organised in each of the sample tables.

**Answer:**

In `table1` each row is a (country, year) with variables cases and population.

```r
## A tibble: 6 x 4
##  country year cases population
##  <chr> <int> <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
```

In `table2`, each row is country, year, variable (“cases”, “population”) combination, and there is a count variable with the numeric value of the combination.

```r
## A tibble: 12 x 4
##  country year type count
##  <chr> <int> <chr> <int>
##1 Afghanistan 1999 cases   745
##2 Afghanistan 1999 population 19987071
##3 Afghanistan 2000 cases   2666
```
In table3, each row is a (country, year) combination with the column rate having the rate of cases to population as a character string in the format “cases/rate”.

Table 4 is split into two tables, one table for each variable: table4a is the table for cases, while table4b is the table for population. Within each table, each row is a country, each column is a year, and the cells are the value of the variable for the table.

2.

Compute the rate for table2, and table4a + table4b. You will need to perform four operations:

Extract the number of TB cases per country per year. Extract the matching population per country per year. Divide cases by population, and multiply by 10000. Store back in the appropriate place. Which representation is easiest to work with? Which is hardest? Why?

Answer:

Using some code from Chapter 13. Relational data
table2

```r
## # A tibble: 12 x 4
## #  country      year     type  count
## # <chr>      <int>     <chr> <int>
## 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: 6 x 4
# country  year     type  cases
# <chr>    <int>     <chr> <int>
# 1 Afghanistan 1999 cases 745
# 2 Afghanistan 2000 cases 2666
# 3 Brazil 1999 cases 37737
# 4 Brazil 2000 cases 80488
# 5 China 1999 cases 212258
# 6 China 2000 cases 213766

# A tibble: 6 x 4
# country  year     type  pop
# <chr>    <int>     <chr> <int>
# 1 Afghanistan 1999 population 19987071
# 2 Afghanistan 2000 population 20595360
# 3 Brazil 1999 population 172006362
# 4 Brazil 2000 population 174504898
# 5 China 1999 population 1272915272
# 6 China 2000 population 1280428583

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

table2_new %>% mutate(rate = (cases/pop)*10000) %>%
  select(country, year, rate) %>%
  arrange(year) %>%
  spread(year, rate)

## # A tibble: 3 x 3
## country  `1999` `2000`
## <chr>      <dbl>  <dbl>
## 1 Afghanistan 0.373  1.29
## 2 Brazil      2.19   4.61
## 3 China       1.67   1.67

Using table4a and table4b

table4a

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

table4b

## # A tibble: 3 x 3
## country  `1999` `2000`
## <chr>   <int>  <int>
## 1 Afghanistan  19987071 20595360
## 2 Brazil         172006362 174504898
## 3 China        1272915272 1280428583
table_new2 <- table4a %>% inner_join(table4b, by = c("country"))
table_new2

## # A tibble: 3 x 5
## <chr>   <int>    <int>   <int>   <int>
## 1 Afghanistan 745 2666 19987071 20595360
## 2 Brazil 37737 80488 172006362 174504898
## 3 China 212258 213766 1272915272 1280428583
table_new2a <- table_new2 %>% mutate(rate.1999 = (`1999.x`/`1999.y`)*10000,
  select(country, rate.1999, rate.2000)
table_new2a

## # A tibble: 3 x 3
## country rate.1999 rate.2000
## <chr>      <dbl>      <dbl>
## 1 Afghanistan 0.373      1.29
## 2 Brazil      2.19       4.61
## 3 China       1.67       1.67
12.3.3

4

Tidy the simple tibble below. Do you need to spread or gather it? What are the variables?

**Answer:**

We need to gather the data into two new columns, sex and count.

```r
preg <- tribble(~pregnant, ~male, ~female,
"yes", NA, 10,
"no", 20, 12)
preg
## # A tibble: 2 x 3
## pregnant male female
## <chr> <dbl> <dbl>
## 1 yes NA 10
## 2 no 20 12

preg %>% gather(male, female, key = "sex", value = "count")
## # A tibble: 4 x 3
## pregnant sex count
## <chr> <chr> <dbl>
## 1 yes male NA
## 2 no male 20
## 3 yes female 10
## 4 no female 12
```

12.4.3

1.

What do the extra and fill arguments do in separate()? Experiment with the various options for the following two toy datasets.

```r
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 x 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"))
## # A tibble: 3 x 3
## one two three
## <chr> <chr> <chr>
## 1 a b c
## 2 d e f
## 3 g h i
```
## Warning: Expected 3 pieces. Missing pieces filled with `NA` in 1 rows [2].

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

Examples:

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

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

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

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