# Problem Set 3

Teaching Team

2021

- Due date: Monday, September 20
- Submission process: Please submit your assignment directly to Gradescope. You can do this by knitting your file and downloading the PDF to your computer. Then navigate to Gradescope.com or via the link on BCourses to submit your assignment.

Helpful hints:

- Knit your file early and often to minimize knitting errors! If you copy and paste code from the slides, you are bound to get an error that is hard to diagnose. Typing out the code is the way to smooth knitting. We recommend knitting your file each time after you write a few sentences/add a new code chunk, so you can detect the source of the knitting error more easily. This will save you and the teaching team time!
- Please make sure that your code does not run off the page of the knitted PDF. If it does, we can't see your work. To avoid this, have a look at your knitted PDF and ensure all the code fits in the file. When it doesn't, go back to your .Rmd file and add spaces (new lines) using the return or enter key so that the code runs onto the next line.

---

### Question 1

In this question you will create a data frame. Below is code for how to do this:

```
# data frame with three columns and three rows
# notice how we start with the column name and then the row values
# each column, in this example, has three values
df_example <- data.frame("column_1" = 1:3,
                         "column_2" = c("string_1", "string_2", "2"),
                         # here we add an NA value due to missing data
                         "column_3" = c(NA, "string_3", 50))
df_example
```

```
##   column_1 column_2 column_3
## 1        1 string_1     <NA>
## 2        2 string_2 string_3
## 3        3        2       50
```

Now you try! Create a data frame with three columns and the following values:

- Column 1: ID

  1, 2, 3
- Column 2: NAME

  "Pam", "Jim", "Dwight"
- Column 3: AGE

  40, NA, 48

```
# your code here
dunder_mifflin <- data.frame("id" = 1:3,
                             "Name" = c("Pam", "Jim", "Dwight"),
                             "Age" = c(40, NA, 48))
dunder_mifflin
```

```
##   id   Name Age
## 1  1    Pam  40
## 2  2    Jim  NA
## 3  3 Dwight  48
```

### Question 2

With your new data frame created in the previous question, find the following values: - length - typeof - class

```
# your code here
length(dunder_mifflin) # 3
```

```
## [1] 3
```

```
typeof(dunder_mifflin) # list
```

```
## [1] "list"
```

```
class(dunder_mifflin) # data.frame
```

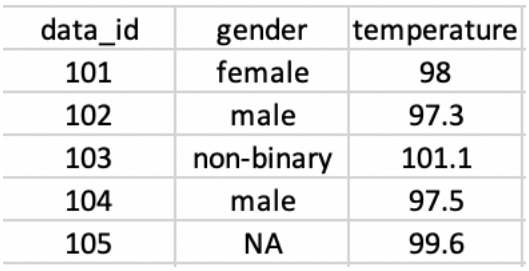
```
## [1] "data.frame"
```

**Question 3**

Create a data frame and a tibble that matches the image below:

```
# by the way, you can load images into rmarkdown! Cool, right?!
# here we use the knitr library (though there are multiple ways to load images)
library(knitr)

# notice that we specify the path to look within the current directory
# by using the period: .
# followed by a slash: / to pull the image file
knitr::include_graphics('./table_replicate.png')
```



| data_id | gender | temperature |
|---|---|---|
| 101 | female | 98 |
| 102 | male | 97.3 |
| 103 | non-binary | 101.1 |
| 104 | male | 97.5 |
| 105 | NA | 99.6 |

Hint: You may need to load a library for tibbles.

```
# your code here
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
df <- data.frame(
  id = 101:105,
  gender = c("female", "male", "non-binary", "male", NA),
  temperature = c(98, 97.3, 101.1, 97.5, 99.6))
df
```

```
##    id     gender temperature
## 1 101     female        98.0
## 2 102       male        97.3
## 3 103 non-binary       101.1
## 4 104       male        97.5
## 5 105       <NA>        99.6
```

```r
tib <- tibble(
  id = 101:105,
  gender = c("female", "male", "non-binary", "male", NA),
  temperature = c(98, 97.3, 101.1, 97.5, 99.6))
tib
```

```
## # A tibble: 5 x 3
##      id gender     temperature
##   <int> <chr>            <dbl>
## 1   101 female            98
## 2   102 male              97.3
## 3   103 non-binary       101.
## 4   104 male              97.5
## 5   105 <NA>              99.6
```

**Question 4**

What are the key differences between data frames and tibbles?

Tibbles are a modern re-imagining of the data frame. The main differences are how tibbles print and subset.

When you print a tibble, it only shows the first 10 rows and as many columns as will fit on your screen. It will also show you the type of each column, which doesn't happen when you print a normal data frame (you would have to use str() to get that information).

Tibbles are more strict about subsetting, so using single square brackets [] will always return another tibble and using double square brackets [[]] will always return a vector. For example, tib[1,2] will return a tibble containing the element located at row 1, column 2 in tib. tib[[1,2]] will return a vector of that element.

```
tib[1,2]
```

```
## # A tibble: 1 x 1
##   gender
##   <chr>
## 1 female
```

```
tib[[1,2]]
```

```
## [1] "female"
```

Why are tibbles preferable?

Tibbles don't do a few things many users find annoying about data frames:

- Tibbles don't change the type of inputs (e.g. strings to factors)
- Tibbles don't change the names of variables
- Tibbles never create row.names()

Tibbles have friendly features baked in, but you may encounter bugs with the tibble format and older packages that haven't been updated. You change a tibble into a data frame with as.data.frame().

To learn more about tibbles click here and/or run vignette("tibble").

**Question 5**

We just found out results for COVID testing and want to add it to our data. Using the tibble you created in Question 3, add the following test results to a new column called "results".

- 101 = NEGATIVE
- 102 = POSITIVE
- 103 = NEGATIVE
- 104 = NEGATIVE
- 105 = NEGATIVE

```
# your code here
tib$results <- c("NEGATIVE", "POSITIVE", "NEGATIVE", "NEGATIVE", "NEGATIVE")
tib
```

```
## # A tibble: 5 x 4
##      id gender     temperature results
##   <int> <chr>            <dbl> <chr>
## 1   101 female            98   NEGATIVE
## 2   102 male              97.3 POSITIVE
## 3   103 non-binary       101.  NEGATIVE
## 4   104 male              97.5 NEGATIVE
## 5   105 <NA>              99.6 NEGATIVE
```

### Question 6

You find out there was an error in data collection and subject 102's temperature is actually 98.3, not 97.3. Correct the value in your data frame.

```
# your code here
tib[2, 3] <- 98.3
tib
```

```
## # A tibble: 5 x 4
##      id gender     temperature results
##   <int> <chr>            <dbl> <chr>
## 1   101 female            98   NEGATIVE
## 2   102 male              98.3 POSITIVE
## 3   103 non-binary       101.  NEGATIVE
## 4   104 male              97.5 NEGATIVE
## 5   105 <NA>              99.6 NEGATIVE
```

### Question 7

Load the "stds-by-disease-county-year-sex.csv" data set, which is in the data folder.

You can find more information about this data set from the California Open Data Portal:

https://data.ca.gov/dataset/stds-in-california-by-disease-county-year-and-sex

```
library(readr)

# your code here
stds <- read_csv("./data/stds-by-disease-county-year-sex.csv",
                 skip = 3)
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Disease = col_character(),
##   County = col_character(),
##   Year = col_double(),
##   Sex = col_character(),
##   Cases = col_double(),
##   Population = col_double()
## )
```

You may have noticed that there are empty cells in the first three rows. Modify your code above (if you haven't already) to remove these rows.

### Question 8

Let's explore this STD data set. **Use code** to find the values requested below. Insert R chunks as needed.

How many rows?

```
str(stds)
```

```
## tibble [9,558 x 6] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Disease   : chr [1:9558] "Chlamydia" "Chlamydia" "Chlamydia" "Chlamydia" ...
##  $ County    : chr [1:9558] "California" "California" "California" "California" ...
##  $ Year      : num [1:9558] 2001 2001 2001 2002 2002 ...
##  $ Sex       : chr [1:9558] "Female" "Male" "Total" "Female" ...
##  $ Cases     : num [1:9558] 75941 24885 101590 81583 28521 ...
##  $ Population: num [1:9558] 17339700 17173042 34512742 17554666 17383624 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Disease = col_character(),
##   ..   County = col_character(),
##   ..   Year = col_double(),
##   ..   Sex = col_character(),
##   ..   Cases = col_double(),
##   ..   Population = col_double()
##   .. )
```

9558

How many columns?

6

What are the column names?

Disease, County, Year, Sex, Cases, Population

What are the column types?

chr, chr, num, chr, num, num

### Question 9

You want to dig deeper into the data and focus on the years 2015 - 2018. Use the which() function to index which rows fit this year range and assign the results to a new data frame. To check whether this was done correctly you should expect the following dimensions: 2124 rows x 6 columns

```
# your code here
(stds_2008_2015 <- stds[which(stds$Year > 2014), ])
```

```
## # A tibble: 2,124 x 6
##    Disease   County      Year Sex     Cases Population
##    <chr>     <chr>      <dbl> <chr>   <dbl>      <dbl>
##  1 Chlamydia California  2015 Female 121749   19634752
##  2 Chlamydia California  2015 Male    67694   19441376
##  3 Chlamydia California  2015 Total  189747   39076128
##  4 Chlamydia California  2016 Female 123924   19758238
##  5 Chlamydia California  2016 Male    73708   19570099
##  6 Chlamydia California  2016 Total  198245   39328337
##  7 Chlamydia California  2017 Female 134847   19891334
##  8 Chlamydia California  2017 Male    83203   19719222
##  9 Chlamydia California  2017 Total  218519   39610556
## 10 Chlamydia California  2018 Female 142397   19989903
## # ... with 2,114 more rows
```

```
dim(stds_2008_2015)
```

```
## [1] 2124    6
```

### Question 10

Your colleague is interested in this data set but hasn't setup their git repository. They ask you to help them out by exporting this new data set as a .csv file. Place your output in the /data folder.

As a test, you can try to read in the .csv you created to make sure everything looks correct.

```
# your code here
write_csv(stds_2008_2015, "data/stds_2008_2015.csv")

import_test <- read_csv("data/stds_2008_2015.csv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Disease = col_character(),
##   County = col_character(),
##   Year = col_double(),
##   Sex = col_character(),
##   Cases = col_double(),
##   Population = col_double()
## )
```

```
import_test
```

```
## # A tibble: 2,124 x 6
##    Disease   County      Year Sex     Cases Population
##    <chr>     <chr>      <dbl> <chr>   <dbl>      <dbl>
##  1 Chlamydia California  2015 Female 121749   19634752
##  2 Chlamydia California  2015 Male    67694   19441376
##  3 Chlamydia California  2015 Total  189747   39076128
##  4 Chlamydia California  2016 Female 123924   19758238
##  5 Chlamydia California  2016 Male    73708   19570099
##  6 Chlamydia California  2016 Total  198245   39328337
##  7 Chlamydia California  2017 Female 134847   19891334
##  8 Chlamydia California  2017 Male    83203   19719222
##  9 Chlamydia California  2017 Total  218519   39610556
## 10 Chlamydia California  2018 Female 142397   19989903
## # ... with 2,114 more rows
```

### Question 11

Look up how to use the unique() function and run it on the County column of the STD data set. You should see a total of 59 counties.

```
# your code here
unique(stds$County)
```

```
##  [1] "California"      "Alameda"         "Alpine"          "Amador"
##  [5] "Butte"           "Calaveras"       "Colusa"          "Contra Costa"
##  [9] "Del Norte"       "El Dorado"       "Fresno"          "Glenn"
## [13] "Humboldt"        "Imperial"        "Inyo"            "Kern"
## [17] "Kings"           "Lake"            "Lassen"          "Los Angeles"
## [21] "Madera"          "Marin"           "Mariposa"        "Mendocino"
## [25] "Merced"          "Modoc"           "Mono"            "Monterey"
## [29] "Napa"            "Nevada"          "Orange"          "Placer"
## [33] "Plumas"          "Riverside"       "Sacramento"      "San Benito"
## [37] "San Bernardino"  "San Diego"       "San Francisco"   "San Joaquin"
## [41] "San Luis Obispo" "San Mateo"       "Santa Barbara"   "Santa Clara"
## [45] "Santa Cruz"      "Shasta"          "Sierra"          "Siskiyou"
## [49] "Solano"          "Sonoma"          "Stanislaus"      "Sutter"
## [53] "Tehama"          "Trinity"         "Tulare"          "Tuolumne"
## [57] "Ventura"         "Yolo"            "Yuba"
```

You decide to focus on one county. Subset your data for one county of your choice.

```
# your code here

# method 1: which
stds_subset <- stds[which(stds$County == "Alameda"), ]
stds_subset
```

```
## # A tibble: 162 x 6
##    Disease   County   Year Sex    Cases Population
##    <chr>     <chr>   <dbl> <chr>  <dbl>      <dbl>
##  1 Chlamydia Alameda  2001 Female  3691     746596
##  2 Chlamydia Alameda  2001 Male    1126     718968
##  3 Chlamydia Alameda  2001 Total   4861    1465564
##  4 Chlamydia Alameda  2002 Female  3729     747987
##  5 Chlamydia Alameda  2002 Male    1126     720481
##  6 Chlamydia Alameda  2002 Total   4870    1468468
##  7 Chlamydia Alameda  2003 Female  3780     747441
##  8 Chlamydia Alameda  2003 Male    1143     719746
##  9 Chlamydia Alameda  2003 Total   4928    1467187
## 10 Chlamydia Alameda  2004 Female  3995     746723
## # ... with 152 more rows
```

```
# method 2: subset function
stds_subset2 <- subset(stds, County == "Alameda")
stds_subset2
```

```
## # A tibble: 162 x 6
```

```
##    Disease   County   Year Sex    Cases Population
##    <chr>     <chr>   <dbl> <chr>  <dbl>      <dbl>
##  1 Chlamydia Alameda  2001 Female  3691     746596
##  2 Chlamydia Alameda  2001 Male    1126     718968
##  3 Chlamydia Alameda  2001 Total   4861    1465564
##  4 Chlamydia Alameda  2002 Female  3729     747987
##  5 Chlamydia Alameda  2002 Male    1126     720481
##  6 Chlamydia Alameda  2002 Total   4870    1468468
##  7 Chlamydia Alameda  2003 Female  3780     747441
##  8 Chlamydia Alameda  2003 Male    1143     719746
##  9 Chlamydia Alameda  2003 Total   4928    1467187
## 10 Chlamydia Alameda  2004 Female  3995     746723
## # ... with 152 more rows
```

```
# method 3: filter function (tidyverse - dplyr)
stds_subset3 <- filter(stds, County == "Alameda")
stds_subset3
```

```
## # A tibble: 162 x 6
##    Disease   County   Year Sex    Cases Population
##    <chr>     <chr>   <dbl> <chr>  <dbl>      <dbl>
##  1 Chlamydia Alameda  2001 Female  3691     746596
##  2 Chlamydia Alameda  2001 Male    1126     718968
##  3 Chlamydia Alameda  2001 Total   4861    1465564
##  4 Chlamydia Alameda  2002 Female  3729     747987
##  5 Chlamydia Alameda  2002 Male    1126     720481
##  6 Chlamydia Alameda  2002 Total   4870    1468468
##  7 Chlamydia Alameda  2003 Female  3780     747441
##  8 Chlamydia Alameda  2003 Male    1143     719746
##  9 Chlamydia Alameda  2003 Total   4928    1467187
## 10 Chlamydia Alameda  2004 Female  3995     746723
## # ... with 152 more rows
```

### Question 12

You're very interested in finding the rate of cases per 100,000 population. In your subset data frame (from the previous question), create a new column called "rate" with the calculated values.

Rate = (Cases / Population) * 100,000

Hint: R allows you to use manipulate variables within a data frame to calculate new values so long as the rows and data types match up. For example: df$var3 <- df$var1 + df$var2

```
# your code here
stds_subset$Rate <- (stds_subset$Cases / stds_subset$Population) * 100000
head(stds_subset)
```

```
## # A tibble: 6 x 7
##   Disease   County   Year Sex    Cases Population  Rate
##   <chr>     <chr>   <dbl> <chr>  <dbl>      <dbl> <dbl>
## 1 Chlamydia Alameda  2001 Female  3691     746596  494.
## 2 Chlamydia Alameda  2001 Male    1126     718968  157.
## 3 Chlamydia Alameda  2001 Total   4861    1465564  332.
## 4 Chlamydia Alameda  2002 Female  3729     747987  499.
## 5 Chlamydia Alameda  2002 Male    1126     720481  156.
## 6 Chlamydia Alameda  2002 Total   4870    1468468  332.
```

You're done! Please knit to pdf and upload to gradescope.