Homework 5

[Ra-Zakee Muhammad]

Due 10/12/2021

Classmates/other resources consulted: [type answer here]

```
library(tidyverse)
```

Question 1 (12 points)

** In each part, say whether the data is tidy or not, and explain why. Hint: only one of them is tidy.**

a.

```
baseball
```

```
## # A tibble: 10 x 4
##    Team_Abbreviation Team_Name          Division `Win-Loss Record`
##    <chr>             <chr>              <chr>    <chr>
##  1 TB                Tampa Bay Rays     East     100-62
##  2 BOS               Boston Red Sox     East     92-70
##  3 NYY               New York Yankees   East     92-70
##  4 TOR               Toronto Blue Jays  East     91-71
##  5 BAL               Baltimore Orioles  East     52-110
##  6 CHW               Chicago White Sox  Central  93-69
##  7 CLE               Cleveland Indians  Central  80-82
##  8 DET               Detroit Tigers     Central  77-85
##  9 KC                Kansas City Royals Central  74-88
## 10 MIN               Minnesota Twins    Central  73-89
```

Not tidy because there are two piece of information placed in the win-loss record column where it could be spereated into two seperate columns

b.

```
stud_fav_colors
```

```
## # A tibble: 5 x 8
##   College                    Red Green  Blue Purple Orange Yellow Other
##   <chr>                    <dbl> <dbl> <dbl>  <dbl>  <dbl>  <dbl> <dbl>
## 1 Claremont McKenna College    36    75    67     24     57     98    99
```

```
## 2 Harvey Mudd College          84    72    19     45     92     54    11
## 3 Pitzer College               36    76    21     33     23     56    44
## 4 Pomona College               98    56    45     32     47     56    88
## 5 Scripps College              34    56    28     73     49     87    33
```

no the rows should be individual students and the college column should be a factors where each student is either from cmc, hmc,scripps, pom, or pitz, and favorite color should be a single column

c.

```
stud_info
```

```
## # A tibble: 10 x 4
##    Name      College Info         Value
##    <chr>     <chr>   <chr>        <dbl>
##  1 Student A CMC     GPA           3.8
##  2 Student B CMC     GPA           3.7
##  3 Student C Pitzer  GPA           3.72
##  4 Student D CMC     GPA           3.66
##  5 Student E Scripps GPA           3.72
##  6 Student A CMC     Graduation 2022
##  7 Student B CMC     Graduation 2024
##  8 Student C Pitzer  Graduation 2023
##  9 Student D CMC     Graduation 2023
## 10 Student E Scripps Graduation 2023
```

no because the info column and value columns do not contain a consistent type of information but rather two different types of info for each column.

d.

```
weather_forecast
```

```
## # A tibble: 7 x 5
##   Day       Temperature_F Wind_mph UV_index ChanceOfRain_percent
##   <chr>             <dbl>    <dbl>    <dbl>                <dbl>
## 1 Thursday             71        7        2                   70
## 2 Friday               63       12        4                   80
## 3 Saturday             71       10        7                    4
## 4 Sunday               78       11        7                    0
## 5 Monday               71       13        6                    7
## 6 Tuesday              70       11        6                    0
## 7 Wednesday            74       10        6                    0
```

yes because each column only contains one type of information and the rows are observational units.

Question 2 (12 points)

For each of these tibbles, perform the necessary operation to make it tidy.

a.

```
avg_weather
```

```
## # A tibble: 12 x 3
##    month     metric           average
##    <chr>     <chr>              <dbl>
##  1 September high_temperature   89
##  2 September low_temperature    60
##  3 September rain_inches         0.15
##  4 September daylight_hours     12.5
##  5 October   high_temperature   80
##  6 October   low_temperature    55
##  7 October   rain_inches         1.05
##  8 October   daylight_hours     11.5
##  9 November  high_temperature   74
## 10 November  low_temperature    47
## 11 November  rain_inches         1.62
## 12 November  daylight_hours     10.5
```

pivot wider by metric

```
avg_weather %>% pivot_wider(names_from = metric, values_from = average)
```

```
## # A tibble: 3 x 5
##   month     high_temperature low_temperature rain_inches daylight_hours
##   <chr>                <dbl>           <dbl>       <dbl>          <dbl>
## 1 September               89              60        0.15           12.5
## 2 October                 80              55        1.05           11.5
## 3 November                74              47        1.62           10.5
```

b.

```
chemicals
```

```
## # A tibble: 6 x 2
##   Chemical_Name Safe_Temperature_Range
##   <chr>         <chr>
## 1 Chemical 1    32-212
## 2 Chemical 2    50-100
## 3 Chemical 3    45-48
## 4 Chemical 4    40-345
## 5 Chemical 5    100-250
## 6 Chemical 6    112-140
```

break up safe temp into low and high

```
chemicals %>% separate(col = Safe_Temperature_Range, into = c("Temp_Low", "Temp_High"), sep = "-")
```

```
## # A tibble: 6 x 3
##   Chemical_Name Temp_Low Temp_High
##   <chr>         <chr>    <chr>
```

```
## 1 Chemical 1    32       212
## 2 Chemical 2    50       100
## 3 Chemical 3    45       48
## 4 Chemical 4    40       345
## 5 Chemical 5    100      250
## 6 Chemical 6    112      140
```

c.

```
cake_prefs
```

```
## # A tibble: 4 x 5
##   class      chocolate vanilla carrot red_velvet
##   <chr>          <dbl>   <dbl>  <dbl>      <dbl>
## 1 Freshmen          34      12     15         32
## 2 Sophomores        23      13     22         29
## 3 Juniors           21      17     18         17
## 4 Seniors           22      33     16         11
```

beak into individual students as observation

```
library(splitstackshape)
```

```
cake_prefs %>%
  pivot_longer(cols = c(chocolate, vanilla, carrot, red_velvet), names_to = "favorite_flavor") %>%
  expandRows("value") %>%
  mutate(Student_index = seq(1:335), Class = class, Favorite_Flavor = favorite_flavor) %>%
  select(Student_index, Class, Favorite_Flavor)
```

```
## # A tibble: 335 x 3
##    Student_index Class    Favorite_Flavor
##            <int> <chr>    <chr>
##  1             1 Freshmen chocolate
##  2             2 Freshmen chocolate
##  3             3 Freshmen chocolate
##  4             4 Freshmen chocolate
##  5             5 Freshmen chocolate
##  6             6 Freshmen chocolate
##  7             7 Freshmen chocolate
##  8             8 Freshmen chocolate
##  9             9 Freshmen chocolate
## 10            10 Freshmen chocolate
## # ... with 325 more rows
```

Question 3 (6 points)

In the U.S., mailing addresses have zipcodes consisting of five digits, then a dash, then four digits. An example might be 91711-4285. Suppose you have a tibble, like the following example, where the first five digits are in a different column than the last four digits.

```
zip_codes
```

```
## # A tibble: 7 x 2
##     Zip PlusFour
##   <dbl>    <dbl>
## 1 91711     3452
## 2 20322     3009
## 3 93782     8473
## 4 78392     8762
## 5 87639     2563
## 6 47628     5416
## 7 20874     5726
```

a. **Use the unite function to replace the two separate columns with a single column consisting of the entire zip code, in the correct format.**

```
zip_codes %>% unite(PlusFour, sep = "-")
```

```
## # A tibble: 7 x 1
##   PlusFour
##   <chr>
## 1 91711-3452
## 2 20322-3009
## 3 93782-8473
## 4 78392-8762
## 5 87639-2563
## 6 47628-5416
## 7 20874-5726
```

b. **Produce the same tibble as in the previous part, but instead of using unite use str_c and any other necessary data transformation function(s).**

mutate and select

```
zip_codes %>% mutate(Zip = str_c(Zip, PlusFour, sep = "-")) %>% select(Zip)
```

```
## # A tibble: 7 x 1
##   Zip
##   <chr>
## 1 91711-3452
## 2 20322-3009
## 3 93782-8473
## 4 78392-8762
## 5 87639-2563
## 6 47628-5416
## 7 20874-5726
```

Question 4 (3 points)

For the nz_cards data set ("New_Zeland_Electronic_card_transactions_aug_2021.csv"), we saw in a previous homework one way to divide the Period column into a year and month using floor(). Here, use a function we've learned this week to divide the Period column into a year column and a month column. Make sure these new columns have the correct data type.

```
nz_cards <- read_csv("New_Zeland_Electronic_card_transactions_aug_2021.csv")
```

```
## Rows: 18024 Columns: 14

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (9): Series_reference, Suppressed, STATUS, UNITS, Subject, Group, Series...
## dbl (3): Period, Data_value, Magnitude
## lgl (2): Series_title_4, Series_title_5

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
nz_cards %>% separate(Period, into = c("Year", "Month"), convert = TRUE)
```

```
## # A tibble: 18,024 x 15
##    Series_reference  Year Month Data_value Suppressed STATUS UNITS   Magnitude
##    <chr>            <int> <int>      <dbl> <chr>      <chr>  <chr>       <dbl>
##  1 ECTA.S19A1        2001     3      2462. <NA>       F      Dollars         6
##  2 ECTA.S19A1        2002     3     17177. <NA>       F      Dollars         6
##  3 ECTA.S19A1        2003     3     22530. <NA>       F      Dollars         6
##  4 ECTA.S19A1        2004     3     28005. <NA>       F      Dollars         6
##  5 ECTA.S19A1        2005     3     30630. <NA>       F      Dollars         6
##  6 ECTA.S19A1        2006     3     33317. <NA>       F      Dollars         6
##  7 ECTA.S19A1        2007     3     36422  <NA>       F      Dollars         6
##  8 ECTA.S19A1        2008     3     39198  <NA>       F      Dollars         6
##  9 ECTA.S19A1        2009     3     40629. <NA>       F      Dollars         6
## 10 ECTA.S19A1        2010     3     41815. <NA>       F      Dollars         6
## # ... with 18,014 more rows, and 7 more variables: Subject <chr>, Group <chr>,
## #   Series_title_1 <chr>, Series_title_2 <chr>, Series_title_3 <chr>,
## #   Series_title_4 <lgl>, Series_title_5 <lgl>
```

Question 5 (3 points)

Look up what the tidyverse's spread() and gather() functions do. These functions are no longer under active development, but exist in a lot of previously written code. Which functions we've learned recently are the updated versions of spread and gather?

Development on spread() is complete, and for new code we recommend switching to pivot_wider() Development on gather() is complete, and for new code we recommend switching to pivot_longer()

Question 6 (3 points)

Why doesn't the following code work as expected? Explain what went wrong here, and why the pivot_wider function doesn't work for this data set in the same way we learned in class.

```
people <- tribble(
  ~name,             ~names,  ~values,
  "Phillip Woods",   "age",       45,
  "Phillip Woods",   "height",   186,
  "Phillip Woods",   "age",       50,
  "Jessica Cordero", "age",       37,
  "Jessica Cordero", "height",   156
)
people %>% pivot_wider(names_from = names, values_from = values)
```

```
## Warning: Values are not uniquely identified; output will contain list-cols.
## * Use `values_fn = list` to suppress this warning.
## * Use `values_fn = length` to identify where the duplicates arise
## * Use `values_fn = {summary_fun}` to summarise duplicates
```

```
## # A tibble: 2 x 3
##   name            age       height
##   <chr>           <list>    <list>
## 1 Phillip Woods   <dbl [2]> <dbl [1]>
## 2 Jessica Cordero <dbl [1]> <dbl [1]>
```

it works if you look at the output it just doesnt give what ur looking 4

Question 7 (12 points)

In all parts below, your examples should be different from the examples we discussed in class.

a. **Give an example of when it's a good idea to replace all NA values in a tibble with 0, and give an example of when this is a bad idea. Explain your answers.**

good - when N/A is in a count factor like number of red crayons in a box, in which case 0 reds is equivalent. bad - when you are taking the mean of a factor like temperature and a day's temp isnt recorded, this doesnt imply 0 degrees.

b. **Give an example of when it's a good idea to use the fill() command to fill in NA values in a tibble, and give an example of when this is a bad idea. Explain your answers.**

good - when multiple rows with missing values are assigned to a common observational unit bad - when an observational unit is missing but the data assigned to it is present in which case the unit above the missing unit will fill in the missing unit and it will had more information assigned to it than necessary.

c. **Give an example of when it's a good idea to use the complete() command on your tibble, and give an example of when this is a bad idea. Explain your answers.**

good - when an observational unit is supposed to have n rows of data assigned to it but less than n rows are present in the table and we want to standardize the appearance of the table.

bad - when there is only one row of data per observational unit in which case using the complete command on any two factos creates unnecessary rows in the tibble.

Question 8 (8 points)

a. (2 points) **Give some examples of data that is implicitly missing from this tibble**

```
campus_visitors
```

```
## # A tibble: 35 x 4
##    School Weekday   TimeOfDay NumVisitors
##    <chr>  <chr>     <chr>           <dbl>
##  1 CMC    Monday    Morning            23
##  2 CMC    Monday    Afternoon          23
##  3 CMC    Monday    Evening            32
##  4 CMC    Tuesday   Morning            42
##  5 CMC    Tuesday   Afternoon          11
##  6 CMC    Tuesday   Evening            12
##  7 CMC    Wednesday Evening             8
##  8 CMC    Thursday  Morning             3
##  9 CMC    Thursday  Afternoon           0
## 10 CMC    Thursday  Evening            14
## # ... with 25 more rows
```

wednesday cmc, friday cmc, monday hmc, tuesday hmc, wednesday hmc,

b. (4 points) **Use the command we learned in class to add rows to this tibble that correspond to the missing data; how many rows do you get? Explain why you get this number.**

```
campus_visitors %>% complete(School, Weekday, TimeOfDay)
```

```
## # A tibble: 60 x 4
##    School Weekday  TimeOfDay NumVisitors
##    <chr>  <chr>    <chr>           <dbl>
##  1 CMC    Friday   Afternoon          35
##  2 CMC    Friday   Evening            NA
##  3 CMC    Friday   Morning             3
##  4 CMC    Monday   Afternoon          23
##  5 CMC    Monday   Evening            32
##  6 CMC    Monday   Morning            23
##  7 CMC    Thursday Afternoon           0
##  8 CMC    Thursday Evening            14
##  9 CMC    Thursday Morning             3
## 10 CMC    Tuesday  Afternoon          11
## # ... with 50 more rows
```

4 colleges * 3 times of day * 5 days of the week = 60 rows

c. (2 points) **Do you think there is still data implicitly missing from the tibble you made in part b? Explain.**

Not implicitly missing because there is a row for every combination of college time of day and day of the week in the table. The entries for scripps college do not appear to be recorded so that could be one instance of implicitly missing data.

Question 9 (4 points)

Explain why a command like "complete(dataset, month, day)" is unlikely to produce the desired result. If you'd like you may reference the following example, though your explanation should be general enough to apply to any data set with a month column and a day column.

```
dataset
```

```
## # A tibble: 42 x 3
##    month     day  value
##    <chr>   <dbl>  <dbl>
##  1 January     1 0.0444
##  2 January     2 0.675
##  3 January     3 0.216
##  4 January     4 0.603
##  5 January     5 0.861
##  6 January     6 0.0995
##  7 January     7 0.696
##  8 January     8 1.21
##  9 January     9 0.121
## 10 January    10 0.453
## # ... with 32 more rows
```

```
dataset %>% complete(month, day)
```

```
## # A tibble: 372 x 3
##    month   day  value
##    <chr> <dbl>  <dbl>
##  1 April     1  0.272
##  2 April     2 NA
##  3 April     3 NA
##  4 April     4 NA
##  5 April     5 NA
##  6 April     6 NA
##  7 April     7 NA
##  8 April     8 NA
##  9 April     9 NA
## 10 April    10 NA
## # ... with 362 more rows
```

This wont give the desired result because all months do not have the same number of days as January so days would be listed that aren't possible on the calendar

Question 10 (6 points)

a. **Create a string in R containing the following sentence, including its punctuation: It's sunny today, but he said, "It'll be rainy tomorrow." To be sure you've made the correct string, print it out using the writeLines function.**

```
string <- "It's sunny today, but he said, \"It'll be rainy tomorrow.\""
writeLines(string)
```

```
## It's sunny today, but he said, "It'll be rainy tomorrow."
```

b. **Explain the *difference between the strings: .* The answer is not just that one has an extra space and one doesn't. Your explanation should mention escape characters.**

The combination of characters backslash and n together is an escape character that starts a new line so that we have "a" on one line an " b" on the next on the other hand the backslash alone with just spaces surrounding it is ignored by the writelines function.

(Note: The strings in this question will cause an error when knitting to PDF; comment them out to compile your document as a PDF. A comment in RMarkdown is:)

Question 11 (6 points)

This question references the following strings

```
s1 <- "the cat, gracie, is hungry"
```

```
s2 <- "The Dog Is Also Hungry!"
```

a. (1 point) **Make s1 uppercase**

```
str_to_upper(s1)
```

```
## [1] "THE CAT, GRACIE, IS HUNGRY"
```

b. (1 point) **Make s2 lowercase**

```
str_to_lower(s2)
```

```
## [1] "the dog is also hungry!"
```

c. (1 point) **Make the first letter of every word in s1 capitalized, while all other letters are lowercase.**

```
s2 %>% str_to_title()
```

```
## [1] "The Dog Is Also Hungry!"
```

d. (3 point) **For s1, carefully use str_sub to capitalize only the first letter of the sentence and the first letter of the cat's name, Gracie.**

```
str_sub(s1, 10, 10) <- "G"
str_sub(s1, 1, 1) <- "T"
s1
```

```
## [1] "The cat, Gracie, is hungry"
```

Question 12 (6 points)

a. **By default, the string_replace_na() function replaces an NA value with the string "NA". How would you modify the following function so that it replaces NA with "n/a", instead of "NA"?**

```
str_replace_na(NA)
```

```
## [1] "NA"
```

```
str_replace_na(NA, replacement = "n/a")
```

```
## [1] "n/a"
```

b. **In the following tibble from the in-class activity, make a new column that concatenates all the relevant info from each meal into a new string. For example, for the row for Friday Breakfast, the new column should have a string that reads "Friday Breakfast: $7". In the row for Monday lunch, the new column should read "Monday Lunch: $0". Unlike in the activity, DO NOT modify any columns of the tibble other than the new one you are making. In particular, the NA's in the Amount column should stay as NA's, and not be converted to anything else.**

```
weekly_meal_spending
```

```
## # A tibble: 21 x 3
##    Day       Meal       Amount
##    <chr>     <chr>       <dbl>
##  1 Friday    Breakfast       7
##  2 Friday    Dinner         25
##  3 Friday    Lunch          17
##  4 Monday    Breakfast      10
##  5 Monday    Dinner         18
##  6 Monday    Lunch          NA
```

```
##  7 Saturday Breakfast     NA
##  8 Saturday Dinner        NA
##  9 Saturday Lunch         19
## 10 Sunday   Breakfast     NA
## # ... with 11 more rows
```

```
weekly_meal_spending %>% mutate( concatenation_column =ifelse(is.na(Amount),str_c(Day, " ", Meal, ":",
```

```
## # A tibble: 21 x 4
##    Day      Meal      Amount concatenation_column
##    <chr>    <chr>      <dbl> <chr>
##  1 Friday   Breakfast      7 Friday Breakfast: $7
##  2 Friday   Dinner        25 Friday Dinner: $25
##  3 Friday   Lunch         17 Friday Lunch: $17
##  4 Monday   Breakfast     10 Monday Breakfast: $10
##  5 Monday   Dinner        18 Monday Dinner: $18
##  6 Monday   Lunch         NA Monday Lunch: $NA
##  7 Saturday Breakfast     NA Saturday Breakfast: $NA
##  8 Saturday Dinner        NA Saturday Dinner: $NA
##  9 Saturday Lunch         19 Saturday Lunch: $19
## 10 Sunday   Breakfast     NA Sunday Breakfast: $NA
## # ... with 11 more rows
```

Question 13 (3 points)

A CMC student's email address consists of their first initial, their last name, and their two digit graduation year, @ cmc.edu. For example, for a student Alice Smith graduating in 2022, her email address is asmith22@cmc.edu. For the table below, write a command that adds a new column consisting of each student's email address, computed from the values in the other columns.

```
students
```

```
## # A tibble: 10 x 3
##    FirstName LastName Graduation
##    <chr>     <chr>         <dbl>
##  1 Mufasa    Adams          2022
##  2 Sarabi    Baker          2024
##  3 Simba     Clark          2023
##  4 Nala      Davis          2022
##  5 Kiara     Evans          2022
##  6 Kovu      Frank          2023
##  7 Timon     Ghosh          2024
##  8 Pumbaa    Hills          2025
##  9 Rafiki    Irwin          2023
## 10 Shenzi    Jones          2022
```

Alice Smith graduating in 2022, her email address is asmith22@cmc.edu.

```
students %>%
  mutate(emails =
           str_c(str_to_lower(str_sub(FirstName, 1,1)),
                       str_to_lower(LastName),
                       str_sub(as.character(Graduation), 3,4),
                       "@cmc.edu" ))
```

```
## # A tibble: 10 x 4
##    FirstName LastName Graduation emails
##    <chr>     <chr>         <dbl> <chr>
##  1 Mufasa    Adams          2022 madams22@cmc.edu
##  2 Sarabi    Baker          2024 sbaker24@cmc.edu
##  3 Simba     Clark          2023 sclark23@cmc.edu
##  4 Nala      Davis          2022 ndavis22@cmc.edu
##  5 Kiara     Evans          2022 kevans22@cmc.edu
##  6 Kovu      Frank          2023 kfrank23@cmc.edu
##  7 Timon     Ghosh          2024 tghosh24@cmc.edu
##  8 Pumbaa    Hills          2025 phills25@cmc.edu
##  9 Rafiki    Irwin          2023 rirwin23@cmc.edu
## 10 Shenzi    Jones          2022 sjones22@cmc.edu
```

Question 14 (16 points)

Consider the file GPAs.csv. Import this data set, and carefully make it both clean and tidy. There will be several steps involved in this process. (the values were randomly generated and do not reflect actual student grades)

```
GPAs <- read_csv("GPAs.csv")
```

```
## Rows: 19 Columns: 4
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): College, Class
## dbl (2): male_GPA, female_GPA
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
GPAs %>% fill(College) %>% mutate(Class = str_to_title(Class)) %>% complete(College, Class) %>% pivot_l
```

```
## # A tibble: 10 x 5
##    college_and_sex_gpa Freshman Junior Senior Sophomore
##    <chr>                 <dbl>  <dbl>  <dbl>     <dbl>
##  1 CMC_female_GPA         3.69   3.7    3.91      3.75
##  2 CMC_male_GPA           3.76   3.36   3.56      3.33
##  3 HMC_female_GPA         3.03  NA      3.21      3.05
```

```
##  4 HMC_male_GPA           3.72  NA      3.43      3.08
##  5 Pitzer_female_GPA      3.13   3.3    3.36      3.82
##  6 Pitzer_male_GPA        3.15   3.56   3.96      3.94
##  7 Pomona_female_GPA      3.63   3.29   3.08      3.14
##  8 Pomona_male_GPA        3.48   3.99   3.57      3.08
##  9 Scripps_female_GPA     3.25   3.84   3.45      3.86
## 10 Scripps_male_GPA       3.53   3.58   3.88      3.52
```