Assignment 4. Data Wrangling with Dplyr- Jeff Moise

Questions

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

1. Compute the follows using `%>%` operator. **Notice that**

- `x %>%\ f = f(x)`
- `x %>% f %>% g = g(f(x))` and
- `x %>% f(y) = f(x,y)`

a. sin(2019)

```
2019 %>% sin()
```

```
## [1] 0.8644605
```

b. sin(cos(2019))

```
2019 %>%
  sin() %>% cos()
```

```
## [1] 0.6490506
```

c. sin(cos(tan(log(2019))))

```
2019 %>% sin() %>% cos() %>% tan() %>% log()
```

```
## [1] -0.2761391
```

d. $\log_2(2019)$

```
2019 %>% log2()
```

```
## [1] 10.97943
```

2. Fixing the `SEX`, `AGE` and `TRAV_SP` following the steps in Assignment 2 (This time, do it on the entire dataset instead of the sample dataset).

```
path <- "C:/Users/student/Documents/RStudio/c2015.xlsx"
library(readxl)
df=read_excel(path)
class(df)
```

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

```
df<-df %>%
  mutate(SEX=case_when(
    SEX=='Not Rep' ~ 'Female',
    SEX=='Unknown' ~ 'Female',
    SEX=='Male' ~ 'Male'))
table(df$SEX)
```

```
##
## Female   Male
##   1358  52598
```

```
df<-df %>%
  mutate(AGE=case_when(
    AGE=='Less than 1' ~ '0',
    TRUE ~ (AGE)))
df$AGE <- as.numeric(df$AGE)
```

```
## Warning: NAs introduced by coercion
```

```
table(df$AGE)
```

```
##
##    0    1    2    3    4    5    6    7    8    9   10   11   12   13   14
##  337  441  410  418  374  389  395  361  384  426  378  369  401  414  525
##   15   16   17   18   19   20   21   22   23   24   25   26   27   28   29
##  718 1091 1530 1899 1939 1931 2029 2051 1994 1893 1820 1676 1613 1374 1495
##   30   31   32   33   34   35   36   37   38   39   40   41   42   43   44
## 1340 1366 1298 1232 1228 1256 1126 1184 1102 1046 1036 1043 1048 1003 1120
##   45   46   47   48   49   50   51   52   53   54   55   56   57   58   59
## 1072 1071 1045 1033 1068 1147 1231 1128 1175 1172 1155 1095 1080 1003  924
##   60   61   62   63   64   65   66   67   68   69   70   71   72   73   74
##  895  811  806  760  722  679  674  625  701  577  521  492  530  400  382
##   75   76   77   78   79   80   81   82   83   84   85   86   87   88   89
##  401  376  330  323  294  313  261  274  262  249  237  202  187  168  138
##   90   91   92   93   94   95   96   97   98   99  101  103  114
##  109   87   67   51   45   27   16   12    6    2    1    1    3
```

```
df$SEX[df$AGE=='Less than 1']= '0'
df$AGE <- as.numeric(df$AGE)
avgAge<-mean(df$AGE[!is.na(df$AGE)])
df$AGE[is.na(df$AGE)]<- avgAge
sum(is.na(df$AGE))
```

```
## [1] 0
```

```
library(stringr)
df$TRAV_SP[df$TRAV_SP=='Stopped'] <- '0'
df$TRAV_SP[df$TRAV_SP=='Not Rep' | df$TRAV_SP=='Unknown'] <- NA
df$TRAV_SP<- stringr::str_replace(df$TRAV_SP," MPH", "")
df$TRAV_SP <- as.numeric(df$TRAV_SP)
```

```
## Warning: NAs introduced by coercion
```

```
mean(df$TRAV_SP, na.rm = TRUE)
```

```
## [1] 44.53279
```

3. Calculate the average age and average speed of female in the accident happened in the weekend.

```
df2<- df %>%
  mutate(
   timeofweek = case_when(
    DAY_WEEK=="Monday" ~ "weekday",
        DAY_WEEK=="Tuesday" ~ "weekday",
        DAY_WEEK=="Wednesday" ~ "weekday",
        DAY_WEEK=="Thursday" ~ "weekday",
        DAY_WEEK=="Friday" ~ "weekday",
    DAY_WEEK=="Saturday" ~ "weekend",
      DAY_WEEK=="Sunday" ~ "weekend"))

head(df2)
```

```
## # A tibble: 6 x 29
##   STATE ST_CASE VEH_NO PER_NO COUNTY   DAY MONTH  HOUR MINUTE   AGE SEX
##   <chr>   <dbl>  <dbl>  <dbl>  <dbl> <dbl> <chr> <dbl>  <dbl> <dbl> <chr>
## 1 Alab~   10001      1      1    127     1 Janu~     2     40    68 Male
## 2 Alab~   10002      1      1     83     1 Janu~    22     13    49 Male
## 3 Alab~   10003      1      1     11     1 Janu~     1     25    31 Male
## 4 Alab~   10003      1      2     11     1 Janu~     1     25    20 <NA>
## 5 Alab~   10004      1      1     45     4 Janu~     0     57    40 Male
## 6 Alab~   10005      1      1     45     7 Janu~     7      9    24 Male
## # ... with 18 more variables: PER_TYP <chr>, INJ_SEV <chr>,
## #   SEAT_POS <chr>, DRINKING <chr>, YEAR <dbl>, MAN_COLL <chr>,
## #   OWNER <chr>, MOD_YEAR <chr>, TRAV_SP <dbl>, DEFORMED <chr>,
## #   DAY_WEEK <chr>, ROUTE <chr>, LATITUDE <dbl>, LONGITUD <dbl>,
## #   HARM_EV <chr>, LGT_COND <chr>, WEATHER <chr>, timeofweek <chr>
```

```
df2 %>%
  filter(SEX=='Female' & DAY_WEEK=="weekend") %>%
  summarize(mean(TRAV_SP, na.rm=TRUE))
```

```
## # A tibble: 1 x 1
##   `mean(TRAV_SP, na.rm = TRUE)`
##                           <dbl>
## 1                           NaN
```

Notice: These questions are to practice `select_if` *and* `summarise_if`, `summarise_all`*... functions in dplyr Check out the uses of these functions here and here.*

4. Use `select_if` and `is.numeric` functions to create a dataset with only numeric variables. Print out the names of all numeric variables

```
select_if(df2, is.numeric)
```

```
## # A tibble: 80,587 x 12
##    ST_CASE VEH_NO PER_NO COUNTY   DAY  HOUR MINUTE   AGE  YEAR TRAV_SP
##      <dbl>  <dbl>  <dbl>  <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl>   <dbl>
##  1   10001      1      1    127     1     2     40    68  2015      55
##  2   10002      1      1     83     1    22     13    49  2015      70
##  3   10003      1      1     11     1     1     25    31  2015      80
##  4   10003      1      2     11     1     1     25    20  2015      80
##  5   10004      1      1     45     4     0     57    40  2015      75
##  6   10005      1      1     45     7     7      9    24  2015      15
##  7   10005      2      1     45     7     7      9    60  2015      65
##  8   10006      1      1    111     8     9     59    64  2015      45
##  9   10006      1      2    111     8     9     59    17  2015      45
## 10   10007      1      1     89     8    18     33    80  2015      NA
## # ... with 80,577 more rows, and 2 more variables: LATITUDE <dbl>,
## #   LONGITUD <dbl>
```

5. Calculate the mean of all numeric variables using `select_if` and `summarise_all`

```
df2 %>%
select_if(list(is.numeric)) %>%
summarise_all(mean)
```

```
## # A tibble: 1 x 12
##   ST_CASE VEH_NO PER_NO COUNTY   DAY  HOUR MINUTE   AGE  YEAR TRAV_SP
##     <dbl>  <dbl>  <dbl>  <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl>   <dbl>
## 1 275607.   1.39   1.63   91.7  15.5  14.0     NA  39.1  2015      NA
## # ... with 2 more variables: LATITUDE <dbl>, LONGITUD <dbl>
```

6. We can shortcut 3 and 4 by using `summarise_if`: Use `summarise_if` to Calculate the mean of all numeric variables. (You may need to use `na.rm = TRUE` to ignore the NAs)

```
df2 %>%
summarise_if(is.numeric,mean, na.rm=TRUE)
```

```
## # A tibble: 1 x 12
##   ST_CASE VEH_NO PER_NO COUNTY   DAY  HOUR MINUTE   AGE  YEAR TRAV_SP
##     <dbl>  <dbl>  <dbl>  <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl>   <dbl>
## 1 275607.   1.39   1.63   91.7  15.5  14.0   28.4  39.1  2015    44.5
## # ... with 2 more variables: LATITUDE <dbl>, LONGITUD <dbl>
```

7. Use `summarise_if` to calculate the median of all numeric variables.

```
df2 %>%
  summarise_if(is.numeric, median, na.rm=TRUE)
```

```
## # A tibble: 1 x 12
##   ST_CASE VEH_NO PER_NO COUNTY   DAY  HOUR MINUTE   AGE  YEAR TRAV_SP
##     <dbl>  <dbl>  <dbl>  <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl>   <dbl>
## 1  270282      1      1     71    15    15     29    37  2015      50
## # ... with 2 more variables: LATITUDE <dbl>, LONGITUD <dbl>
```

8. Use `summarise_if` to calculate the standard deviation of all numeric variables. (`sd` function for standard deviation)

```
df2 %>%
  summarise_if(is.numeric,sd,na.rm=TRUE)
```

```
## # A tibble: 1 x 12
##   ST_CASE VEH_NO PER_NO COUNTY   DAY  HOUR MINUTE   AGE  YEAR TRAV_SP
##     <dbl>  <dbl>  <dbl>  <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl>   <dbl>
## 1 163031.   1.45   1.84   95.0  8.78  9.06   17.3  20.1     0    25.1
## # ... with 2 more variables: LATITUDE <dbl>, LONGITUD <dbl>
```

9. Use `summarise_if` to calculate the number of missing values for each numeric variables. *Hint:* Use `~sum(is.na(.))`

```
df2 %>%
  summarise_if(is.numeric, ~sum(is.na(.)))
```

```
## # A tibble: 1 x 12
##   ST_CASE VEH_NO PER_NO COUNTY   DAY  HOUR MINUTE   AGE  YEAR TRAV_SP
##     <int>  <int>  <int>  <int> <int> <int>  <int> <int> <int>   <int>
## 1       0      0      0      0     0     0    377     0     0   51420
## # ... with 2 more variables: LATITUDE <int>, LONGITUD <int>
```

10. Calculate the `log` of the average for each numeric variable.

```
df2 %>%
  summarise_if(is.numeric,~log(mean(.)))
```

```
## # A tibble: 1 x 12
##   ST_CASE VEH_NO PER_NO COUNTY   DAY  HOUR MINUTE   AGE  YEAR TRAV_SP
##     <dbl>  <dbl>  <dbl>  <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl>   <dbl>
## 1    12.5  0.329  0.488   4.52  2.74  2.64     NA  3.67  7.61      NA
## # ... with 2 more variables: LATITUDE <dbl>, LONGITUD <dbl>
```

11. You will notice that there is one NA is produced in Fix this by calculating the log of the absolute value average for each numeric variable.

```
df2 %>%
  summarise_if(is.numeric,~log(abs(mean(.))))
```

```
## # A tibble: 1 x 12
##   ST_CASE VEH_NO PER_NO COUNTY   DAY  HOUR MINUTE   AGE  YEAR TRAV_SP
##     <dbl>  <dbl>  <dbl>  <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl>   <dbl>
## 1    12.5  0.329  0.488   4.52  2.74  2.64     NA  3.67  7.61      NA
## # ... with 2 more variables: LATITUDE <dbl>, LONGITUD <dbl>
```

12. Calculate the number of missing values for each categorical variables using `summarise_if`

```
df2 %>%
  summarise_if(is.character, ~sum(is.na(.)))
```

```
## # A tibble: 1 x 17
##   STATE MONTH   SEX PER_TYP INJ_SEV SEAT_POS DRINKING MAN_COLL OWNER
##   <int> <int> <int>   <int>   <int>    <int>    <int>    <int> <int>
## 1     0     0 26631       0       0        0        0     7197  7197
## # ... with 8 more variables: MOD_YEAR <int>, DEFORMED <int>,
## #   DAY_WEEK <int>, ROUTE <int>, HARM_EV <int>, LGT_COND <int>,
## #   WEATHER <int>, timeofweek <int>
```

13. Calculate the number of missing values for each categorical variables using `summarise_all`

```
df2 %>%
select_if(is.character) %>%
summarise_all(~sum(is.na(.)))
```

```
## # A tibble: 1 x 17
##   STATE MONTH   SEX PER_TYP INJ_SEV SEAT_POS DRINKING MAN_COLL OWNER
##   <int> <int> <int>   <int>   <int>    <int>    <int>    <int> <int>
## 1     0     0 26631       0       0        0        0     7197  7197
## # ... with 8 more variables: MOD_YEAR <int>, DEFORMED <int>,
## #   DAY_WEEK <int>, ROUTE <int>, HARM_EV <int>, LGT_COND <int>,
## #   WEATHER <int>, timeofweek <int>
```

14. Calculate the number of states in the dataset. **Hint: You can use `length(table())`

```
df2 %>%
  summarise_all(~length(table(STATE)))
```

```
## # A tibble: 1 x 29
##   STATE ST_CASE VEH_NO PER_NO COUNTY   DAY MONTH  HOUR MINUTE   AGE   SEX
##   <int>   <int>  <int>  <int>  <int> <int> <int> <int>  <int> <int> <int>
## 1    51       1      1      1      1     1     1     1      1     1     1
## # ... with 18 more variables: PER_TYP <int>, INJ_SEV <int>,
## #   SEAT_POS <int>, DRINKING <int>, YEAR <int>, MAN_COLL <int>,
## #   OWNER <int>, MOD_YEAR <int>, TRAV_SP <int>, DEFORMED <int>,
## #   DAY_WEEK <int>, ROUTE <int>, LATITUDE <int>, LONGITUD <int>,
## #   HARM_EV <int>, LGT_COND <int>, WEATHER <int>, timeofweek <int>
```

```
# could also use summarise_at()
df2 %>%
  summarise_at(c("STATE"),~length(table(STATE)))
```

```
## # A tibble: 1 x 1
##   STATE
##   <int>
## 1    51
```

15. Calculate the number of uniques values for each categorical variables using `summarise_if`.

```
df2 %>%
  summarise_if(is.character, ~length(table(.)))
```

```
## # A tibble: 1 x 17
##   STATE MONTH   SEX PER_TYP INJ_SEV SEAT_POS DRINKING MAN_COLL OWNER
##   <int> <int> <int>   <int>   <int>    <int>    <int>    <int> <int>
## 1    51    12     2      11       8       29        4       11     8
## # ... with 8 more variables: MOD_YEAR <int>, DEFORMED <int>,
## #   DAY_WEEK <int>, ROUTE <int>, HARM_EV <int>, LGT_COND <int>,
## #   WEATHER <int>, timeofweek <int>
```

16. Calculate the number of uniques values for each categorical variables using `summarise_all`.

```
df2 %>%
  select_if(is.character) %>%
  summarise_all(~length(table(.)))
```

```
## # A tibble: 1 x 17
##   STATE MONTH   SEX PER_TYP INJ_SEV SEAT_POS DRINKING MAN_COLL OWNER
##   <int> <int> <int>   <int>   <int>    <int>    <int>    <int> <int>
## 1    51    12     2      11       8       29        4       11     8
## # ... with 8 more variables: MOD_YEAR <int>, DEFORMED <int>,
## #   DAY_WEEK <int>, ROUTE <int>, HARM_EV <int>, LGT_COND <int>,
## #   WEATHER <int>, timeofweek <int>
```

17. Print out the names of all variables that have more than 30 distinct values

```
names(df2 %>%
  select_if(~length(table(.))>30))
```

```
##  [1] "STATE"    "ST_CASE"  "VEH_NO"   "PER_NO"   "COUNTY"   "DAY"
##  [7] "MINUTE"   "AGE"      "MOD_YEAR" "TRAV_SP"  "LATITUDE" "LONGITUD"
## [13] "HARM_EV"
```

18. Print out the names of all categorical variables that more than 30 distinct values

```
names(df2 %>% select_if(is.character) %>%
       select_if(~length(table(.))>30))
```

```
## [1] "STATE"    "MOD_YEAR" "HARM_EV"
```

19. Print out the names of all numeric variables that has the maximum values greater than 30

```
names(df2 %>% select_if(is.numeric) %>%
       select_if(~max(table(.))>30))
```

```
##  [1] "ST_CASE"  "VEH_NO"   "PER_NO"   "COUNTY"   "DAY"      "HOUR"
##  [7] "MINUTE"   "AGE"      "YEAR"     "TRAV_SP"  "LATITUDE" "LONGITUD"
```

20. Calculate the mean of all numeric variables that has the maximum values greater than 30 using 'summarise_if'

```
df2 %>% select_if(is.numeric) %>%
       summarize_if(~max(table(.))>30, mean, na.rm=1)
```

```
## # A tibble: 1 x 12
##   ST_CASE VEH_NO PER_NO COUNTY   DAY  HOUR MINUTE   AGE  YEAR TRAV_SP
##     <dbl>  <dbl>  <dbl>  <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl>   <dbl>
## 1 275607.   1.39   1.63   91.7  15.5  14.0   28.4  39.1  2015    44.5
## # ... with 2 more variables: LATITUDE <dbl>, LONGITUD <dbl>
```

21. Calculate the mean of all numeric variables that has the maximum values greater than 30 using 'summarise_all'

```
names(df2 %>% select_if(is.numeric) %>%
        select_if(~max(table(.))>30) %>%
        summarize_all(mean,na.rm=1))
```

```
##  [1] "ST_CASE"  "VEH_NO"   "PER_NO"   "COUNTY"   "DAY"      "HOUR"
##  [7] "MINUTE"   "AGE"      "YEAR"     "TRAV_SP"  "LATITUDE" "LONGITUD"
```

22. Create a dataset containing variables with standard deviation greater than 10. Call this data `d1`

```
d0= names(df2 %>% select_if(is.numeric))[df2 %>% select_if(is.numeric) %>%
  summarise_all(sd, na.rm=1) > 10]

d1 = df2 %>% select(d0)

head(d1)
```

```
## # A tibble: 6 x 6
##   ST_CASE COUNTY MINUTE   AGE TRAV_SP LONGITUD
##     <dbl>  <dbl>  <dbl> <dbl>   <dbl>    <dbl>
## 1   10001    127     40    68      55    -87.3
## 2   10002     83     13    49      70    -86.9
## 3   10003     11     25    31      80    -85.8
## 4   10003     11     25    20      80    -85.8
## 5   10004     45     57    40      75    -85.5
## 6   10005     45      9    24      15    -85.5
```

23. Centralizing a variable is subtract it by its mean. Centralize the variables of `d1` using `mutate_all`. Check the means of all centralized variables to confirm that they are all zeros.

```
d1 %>%
  mutate_all(funs(.-mean(.,na.rm=1))) %>%
  summarize_all(mean, na.rm=1)
```

```
## Warning: funs() is soft deprecated as of dplyr 0.8.0
## Please use a list of either functions or lambdas:
##
##   # Simple named list:
##   list(mean = mean, median = median)
##
##   # Auto named with `tibble::lst()`:
##   tibble::lst(mean, median)
##
##   # Using lambdas
##   list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
## This warning is displayed once per session.
```

```
## # A tibble: 1 x 6
##    ST_CASE   COUNTY    MINUTE      AGE   TRAV_SP  LONGITUD
##      <dbl>    <dbl>     <dbl>    <dbl>     <dbl>     <dbl>
## 1 4.73e-11 1.32e-14 -1.25e-15 1.58e-15 -2.49e-15 -6.92e-15
```

24. Standarizing a variable is to subtract it to its mean and then divide by its standard deviation. Standardize the variables of `d1` using `mutate_all`. Check the means and standard deviation of all centralized variables to confirm that they are all zeros (for the means) and ones (for standard deviation).

```
d1 %>%
  mutate_all(funs(.-mean(.,na.rm=1))) %>%
  mutate_all(funs(./sd(.))) %>%
  summarize_all(c(mean,sd), na.rm=1)
```

```
## # A tibble: 1 x 12
##   ST_CASE_fn1 COUNTY_fn1 MINUTE_fn1  AGE_fn1 TRAV_SP_fn1 LONGITUD_fn1
##         <dbl>      <dbl>      <dbl>    <dbl>       <dbl>        <dbl>
## 1   -9.97e-17   1.15e-16        NaN 8.49e-17         NaN          NaN
## # ... with 6 more variables: ST_CASE_fn2 <dbl>, COUNTY_fn2 <dbl>,
## #   MINUTE_fn2 <dbl>, AGE_fn2 <dbl>, TRAV_SP_fn2 <dbl>, LONGITUD_fn2 <dbl>
```