Homework3

Load the into R and print the first few values of the columns with a header including sleep

```
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
```

```
library(tidyverse)
```

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

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v stringr 1.4.0
## v tidyr   1.1.2     v forcats 0.5.0
## v readr   1.3.1

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

```
library(ggplot2)
library(tidyr)
library(readr)
sleep <- read.csv("msleep_ggplot2.csv")
msleep <- tbl_df(sleep)
```

```
## Warning: `tbl_df()` is deprecated as of dplyr 1.0.0.
## Please use `tibble::as_tibble()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```

(a.)(10 pts) Count the number of animals which weigh under 1 kilogram and sleep more than 14 hours a day. (filter(), query())

```
part_a <- select(msleep, name  , sleep_total , bodywt ) %>%
  filter(msleep$sleep_total>=14, msleep$bodywt <= 1)
count(part_a)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1    14
```

(b.)(10 pts) Print the name, order, sleep total and bodyweight of the animals with the 6 longest sleep times, in order of sleep time. (select(), arrange())

```
part_b <- select(msleep,name,order,sleep_total,sleep_rem,sleep_cycle,bodywt) %>%
  arrange(desc(sleep_total)) %>% head
part_b
```

```
## # A tibble: 6 x 6
##   name                  order          sleep_total sleep_rem sleep_cycle bodywt
##   <chr>                 <chr>                <dbl>     <dbl>       <dbl>  <dbl>
## 1 Little brown bat      Chiroptera            19.9       2          0.2    0.01
## 2 Big brown bat         Chiroptera            19.7       3.9        0.117  0.023
## 3 Thick-tailed opposum  Didelphimorph~        19.4       6.6       NA      0.37
## 4 Giant armadillo       Cingulata             18.1       6.1       NA     60
## 5 North American Opossum Didelphimorph~       18         4.9        0.333  1.7
## 6 Long-nosed armadillo  Cingulata             17.4       3.1        0.383  3.5
```

(c.) Add two new columns to the dataframe; wt_ratio with the ratio of brain size to body weight, rem_ratio with the ratio of rem sleep to sleep time. If you think they might be useful, feel free to extract more features than these, and describe what they are. (mutate(), assign())

```
msleep %>%
  mutate(wt_ratio = msleep$brainwt/msleep$bodywt) %>%
  mutate(rem_ratio= msleep$sleep_rem/msleep$sleep_total)
```

```
## # A tibble: 83 x 13
##    name  genus vore  order conservation sleep_total sleep_rem sleep_cycle awake
##    <chr> <chr> <chr> <chr> <chr>              <dbl>     <dbl>       <dbl> <dbl>
##  1 Chee~ Acin~ carni Carn~ lc                  12.1      NA        NA      11.9
##  2 Owl ~ Aotus omni  Prim~ <NA>                17         1.8      NA       7
##  3 Moun~ Aplo~ herbi Rode~ nt                  14.4       2.4      NA       9.6
##  4 Grea~ Blar~ omni  Sori~ lc                  14.9       2.3       0.133   9.1
##  5 Cow   Bos   herbi Arti~ domesticated         4         0.7       0.667  20
##  6 Thre~ Brad~ herbi Pilo~ <NA>                14.4       2.2       0.767   9.6
##  7 Nort~ Call~ carni Carn~ vu                   8.7       1.4       0.383  15.3
##  8 Vesp~ Calo~ <NA>  Rode~ <NA>                 7        NA        NA      17
##  9 Dog   Canis carni Carn~ domesticated        10.1       2.9       0.333  13.9
## 10 Roe ~ Capr~ herbi Arti~ lc                   3        NA        NA      21
## # ... with 73 more rows, and 4 more variables: brainwt <dbl>, bodywt <dbl>,
## #   wt_ratio <dbl>, rem_ratio <dbl>
```

(d.)Display the average, min and max sleep times for each order. (group_by(), summarise(), groupby(), agg())

```
msleep %>%
  group_by(order) %>%
  summarise(avg=mean(sleep_total),min=min(sleep_total),max=max(sleep_total))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 19 x 4
##    order             avg   min   max
##    <chr>           <dbl> <dbl> <dbl>
##  1 Afrosoricida    15.6   15.6  15.6
##  2 Artiodactyla     4.52   1.9   9.1
##  3 Carnivora       10.1    3.5  15.8
##  4 Cetacea          4.5    2.7   5.6
##  5 Chiroptera      19.8   19.7  19.9
##  6 Cingulata       17.8   17.4  18.1
##  7 Didelphimorphia 18.7   18    19.4
##  8 Diprotodontia   12.4   11.1  13.7
##  9 Erinaceomorpha  10.2   10.1  10.3
## 10 Hyracoidea       5.67   5.3   6.3
## 11 Lagomorpha       8.4    8.4   8.4
## 12 Monotremata      8.6    8.6   8.6
## 13 Perissodactyla   3.47   2.9   4.4
## 14 Pilosa          14.4   14.4  14.4
## 15 Primates        10.5    8    17
## 16 Proboscidea      3.6    3.3   3.9
## 17 Rodentia        12.5    7    16.6
## 18 Scandentia       8.9    8.9   8.9
## 19 Soricomorpha    11.1    8.4  14.9
```

(e.1)Impute the missing brain weights as the average wt_ratio for that animal's order times the animal's weight. . (group_by(), mutate())

```
part_e1<-msleep %>%
  group_by(order) %>%
  mutate(brainwt= ifelse(is.na(brainwt),(mean(brainwt,na.rm =TRUE)/mean(bodywt,na.rm= TRUE))*bodywt,bra
part_e1
```

```
## # A tibble: 83 x 11
## # Groups:   order [19]
##    name  genus vore  order conservation sleep_total sleep_rem sleep_cycle awake
##    <chr> <chr> <chr> <chr> <chr>              <dbl>     <dbl>       <dbl> <dbl>
##  1 Chee~ Acin~ carni Carn~ lc                  12.1      NA        NA      11.9
##  2 Owl ~ Aotus omni  Prim~ <NA>                17         1.8      NA       7
##  3 Moun~ Aplo~ herbi Rode~ nt                  14.4       2.4      NA       9.6
##  4 Grea~ Blar~ omni  Sori~ lc                  14.9       2.3       0.133   9.1
##  5 Cow   Bos   herbi Arti~ domesticated         4         0.7       0.667  20
##  6 Thre~ Brad~ herbi Pilo~ <NA>                14.4       2.2       0.767   9.6
##  7 Nort~ Call~ carni Carn~ vu                   8.7       1.4       0.383  15.3
##  8 Vesp~ Calo~ <NA>  Rode~ <NA>                 7        NA        NA      17
##  9 Dog   Canis carni Carn~ domesticated        10.1       2.9       0.333  13.9
## 10 Roe ~ Capr~ herbi Arti~ lc                   3        NA        NA      21
## # ... with 73 more rows, and 2 more variables: brainwt <dbl>, bodywt <dbl>
```

(e.2) Make a second copy of your dataframe, but this time impute missing brain weights with the average brain weight for that animal's order.

```
part_e2 <- msleep %>%
  group_by(order) %>%
  mutate(brainwt= ifelse(is.na(brainwt),mean(brainwt,na.rm = TRUE),brainwt))
part_e2
```

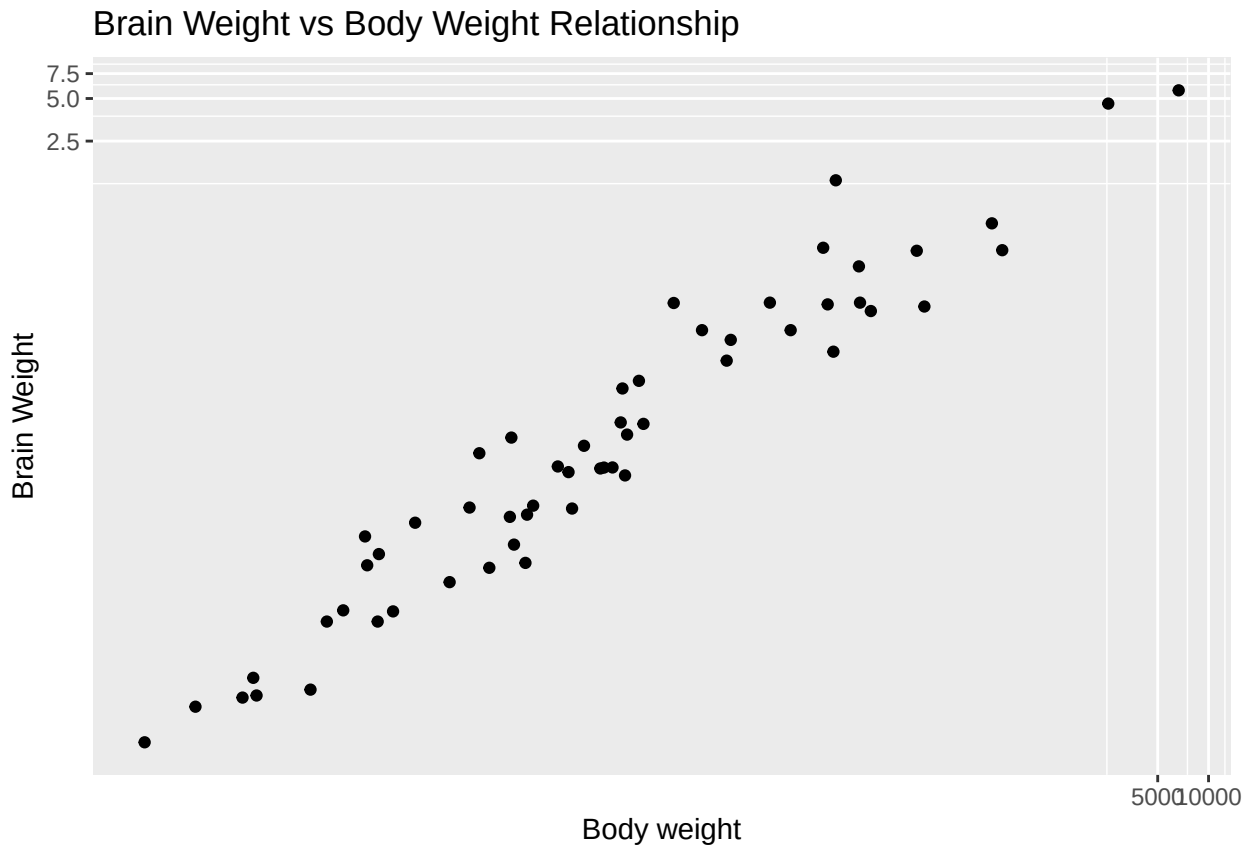
```
## # A tibble: 83 x 11
## # Groups:   order [19]
##    name  genus vore  order conservation sleep_total sleep_rem sleep_cycle awake
##    <chr> <chr> <chr> <chr> <chr>              <dbl>     <dbl>       <dbl> <dbl>
##  1 Chee~ Acin~ carni Carn~ lc                  12.1      NA        NA      11.9
##  2 Owl ~ Aotus omni  Prim~ <NA>                17         1.8      NA       7
##  3 Moun~ Aplo~ herbi Rode~ nt                  14.4       2.4      NA       9.6
##  4 Grea~ Blar~ omni  Sori~ lc                  14.9       2.3       0.133   9.1
##  5 Cow   Bos   herbi Arti~ domesticated         4         0.7       0.667  20
##  6 Thre~ Brad~ herbi Pilo~ <NA>                14.4       2.2       0.767   9.6
##  7 Nort~ Call~ carni Carn~ vu                   8.7       1.4       0.383  15.3
##  8 Vesp~ Calo~ <NA>  Rode~ <NA>                 7        NA        NA      17
##  9 Dog   Canis carni Carn~ domesticated        10.1       2.9       0.333  13.9
## 10 Roe ~ Capr~ herbi Arti~ lc                   3        NA        NA      21
## # ... with 73 more rows, and 2 more variables: brainwt <dbl>, bodywt <dbl>
```

(e.3) What assumptions do these data filling methods make? Which is the best way to impute the data, or do you see a better way, and why? You may impute or remove other variables as you find appropriate. Briefly explain your decisions

In my perspective, imputing the missing weights as the average wt_ratio for animals order * Animals body weight is a better way to impute the data. There is ratio called brain-to-body mass ratio which is termed to be a rough estimate for the depicting the intelligence in an animal. Although this relationship follows separate linear functions for cold-warm blooded animals, brain size usually increases with body size (with exceptions).

```
linear <- ggplot(msleep, aes(x=bodywt ,y =brainwt)) +geom_point() + coord_trans(x="log10",y="log10")
linear <- linear + labs(title = "Brain Weight vs Body Weight Relationship",x="Body weight",y="Brain Wei
show(linear)
```

```
## Warning: Removed 27 rows containing missing values (geom_point).
```



Therefore, the first method would be better to use for missing brain weight values.

Question 2. Grab the dataset from the tidyr package (tidyr::who), and tidy it as shown in the case study before answering the following questions

```
library(dplyr)
library(tidyverse)
library(ggplot2)
library(tidyr)
tidyr::who
```

```
## # A tibble: 7,240 x 60
##    country iso2  iso3   year new_sp_m014 new_sp_m1524 new_sp_m2534 new_sp_m3544
##    <chr>   <chr> <chr> <int>       <int>        <int>        <int>        <int>
##  1 Afghan~ AF    AFG    1980          NA           NA           NA           NA
##  2 Afghan~ AF    AFG    1981          NA           NA           NA           NA
##  3 Afghan~ AF    AFG    1982          NA           NA           NA           NA
##  4 Afghan~ AF    AFG    1983          NA           NA           NA           NA
##  5 Afghan~ AF    AFG    1984          NA           NA           NA           NA
##  6 Afghan~ AF    AFG    1985          NA           NA           NA           NA
##  7 Afghan~ AF    AFG    1986          NA           NA           NA           NA
##  8 Afghan~ AF    AFG    1987          NA           NA           NA           NA
##  9 Afghan~ AF    AFG    1988          NA           NA           NA           NA
## 10 Afghan~ AF    AFG    1989          NA           NA           NA           NA
## # ... with 7,230 more rows, and 52 more variables: 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>
```

```
who %>%
  pivot_longer(
    cols = new_sp_m014:newrel_f65,
    names_to = "key",
    values_to = "cases",
    values_drop_na = TRUE
  ) %>%
  mutate(
    key = stringr::str_replace(key, "newrel", "new_rel")
  ) %>%
  separate(key, c("new", "var", "sexage")) %>%
  select(-new, -iso2, -iso3) %>%
  separate(sexage, c("sex", "age"), sep = 1)
```

```
## # A tibble: 76,046 x 6
##    country      year var   sex   age   cases
##    <chr>       <int> <chr> <chr> <chr> <int>
##  1 Afghanistan  1997 sp    m     014       0
##  2 Afghanistan  1997 sp    m     1524     10
##  3 Afghanistan  1997 sp    m     2534      6
##  4 Afghanistan  1997 sp    m     3544      3
##  5 Afghanistan  1997 sp    m     4554      5
##  6 Afghanistan  1997 sp    m     5564      2
##  7 Afghanistan  1997 sp    m     65        0
##  8 Afghanistan  1997 sp    f     014       5
##  9 Afghanistan  1997 sp    f     1524     38
## 10 Afghanistan  1997 sp    f     2534     36
## # ... with 76,036 more rows
```

(a) Explain why this line > mutate(key = stringr::str_replace(key, “newrel”, “new_rel”))

The above line is used to replace all entries with “newrel” to “new_rel”. This is because the former entries adds to the inconsistency of the dataset. Replacing the entries will substitute the values so all the variable names are unvarying. If we skip this line in the tidying process , “newrel” and “new_rel” will be treated as separate types.Therefore, advancing functions will separate the variables that fit and give errors.

(b) How many entries are removed from the dataset when you set values_drop_na to true in the pivot_longer command (in this dataset)?

```r
who_false <- who %>%
    pivot_longer(
      cols = new_sp_m014:newrel_f65,
      names_to = "key",
      values_to = "cases",
      values_drop_na = FALSE
    ) %>%
    mutate(
      key = stringr::str_replace(key, "newrel", "new_rel")
    ) %>%
    separate(key, c("new", "var", "sexage")) %>%
    select(-new, -iso2, -iso3) %>%
    separate(sexage, c("sex", "age"), sep = 1)
glimpse(who_false)
```

```
## Rows: 405,440
## Columns: 6
## $ country <chr> "Afghanistan", "Afghanistan", "Afghanistan", "Afghanistan",...
## $ year    <int> 1980, 1980, 1980, 1980, 1980, 1980, 1980, 1980, 1980, 1980,...
## $ var     <chr> "sp", "sp", "sp", "sp", "sp", "sp", "sp", "sp", "sp", "sp",...
## $ sex     <chr> "m", "m", "m", "m", "m", "m", "m", "f", "f", "f", "f", "f",...
## $ age     <chr> "014", "1524", "2534", "3544", "4554", "5564", "65", "014",...
## $ cases   <int> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,...
```

In this dataset , 405,440 - 76,046 = 329,394 row entries have been removed.

(c) Explain the difference between an explicit and implicit missing value, in general. Can you find any implicit missing values in this dataset, if so where?

Values are missing in 2 different possible ways: i. Explicit: Such a missing value will be presented with (N/A) denoting that there is an absence of a value.("presence of an absence"). If data representation does not consider these values important, we can turn explicit missing values implicit.

ii. Implicit: Such a missing value will not be appeared in the given dataset.("absence of a presence").We can make changes to the dataset and make implicit missing values explicit.

(d) Looking at the features (country, year, var, sex, age, cases) in the tidied data, are they all appropriately typed? Are there any features you think would be better suited as a different type? Why or why not?

In my perspective,all the variables in the tidied data are appropriately typed except sex where we can change it into a factor-type vector that contains a set of numeric codes with character-valued levels.

```r
who_factor<- who %>%
  pivot_longer(
    cols = new_sp_m014:newrel_f65,
    names_to = "key",
    values_to = "cases",
    values_drop_na = TRUE
  ) %>%
  mutate(
    key = stringr::str_replace(key, "newrel", "new_rel")
```

```
  ) %>%
  separate(key, c("new", "var", "sexage")) %>%
  select(-new, -iso2, -iso3) %>%
  separate(sexage, c("sex", "age"), sep = 1)
who_factor$sex <- as.factor(who_factor$sex)
who_factor
```

```
## # A tibble: 76,046 x 6
##    country      year var   sex   age   cases
##    <chr>       <int> <chr> <fct> <chr> <int>
##  1 Afghanistan  1997 sp    m     014       0
##  2 Afghanistan  1997 sp    m     1524     10
##  3 Afghanistan  1997 sp    m     2534      6
##  4 Afghanistan  1997 sp    m     3544      3
##  5 Afghanistan  1997 sp    m     4554      5
##  6 Afghanistan  1997 sp    m     5564      2
##  7 Afghanistan  1997 sp    m     65        0
##  8 Afghanistan  1997 sp    f     014       5
##  9 Afghanistan  1997 sp    f     1524     38
## 10 Afghanistan  1997 sp    f     2534     36
## # ... with 76,036 more rows
```

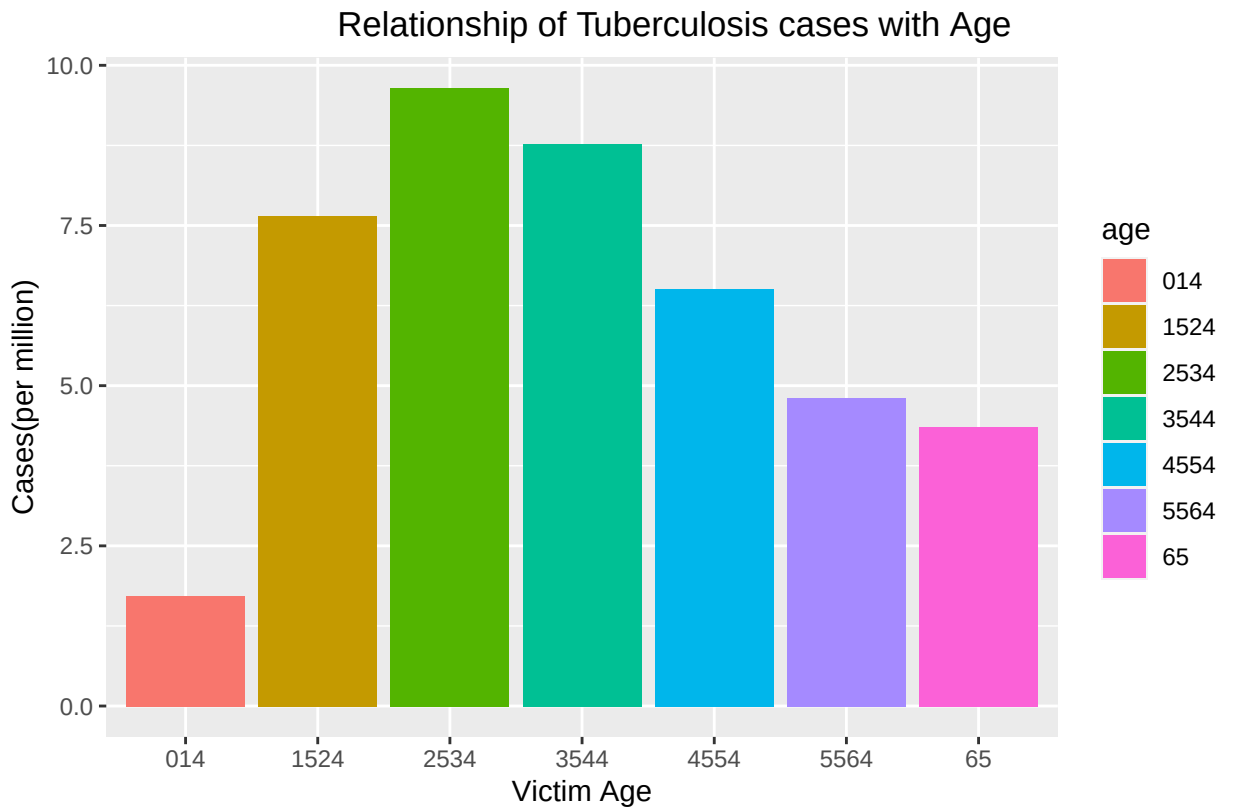
```
mode(who_factor$sex)
```

```
## [1] "numeric"
```

(e) Generate an informative visualization, which shows something about the data. Give a brief description of what it shows, and why you thought it would be interesting to investigate

```
plot <- ggplot(who_factor,aes(x=age,y=cases/1000000,fill=age)) + geom_bar(stat='identity')
plot <- plot + labs(title = "
                        Relationship of Tuberculosis cases with Age"
                    , x = "Victim Age",y="Cases(per million)" )

plot
```



By looking at the plot above, we can infer that TB disease is less likely to occur for individuals aging under 15. Rather this graph denotes individuals aging 25-34 at maximum risk for Tuberculosis.

(f.) Construct the given table and show the code to tidy the dataset.

Step 1: Construct table as it as

```
Site <- c("facebook","myspace","snapchat","twitter","tiktok")
U30.F <- c(30,1,6,18,44)
U30_M <- c(35,2,5,23,60)
O30.F <- c(66,3,3,12,2)
O30.M <- c(58,6,2,28,7)
site_Demo <- data.frame(Site,U30.F,U30_M,O30.F,O30.M)
site_Demo
```

```
##       Site U30.F U30_M O30.F O30.M
## 1 facebook    30    35    66    58
## 2  myspace     1     2     3     6
## 3 snapchat     6     5     3     2
## 4  twitter    18    23    12    28
## 5   tiktok    44    60     2     7
```

Step 2: Tidy the given dataset (using gather()/pivot_longer() and

```
site_tidy <- site_Demo  %>%
  gather("Gender","Count",2:5)
site_tidy
```

```
##        Site Gender Count
## 1  facebook  U30.F    30
## 2   myspace  U30.F     1
## 3  snapchat  U30.F     6
## 4   twitter  U30.F    18
## 5    tiktok  U30.F    44
## 6  facebook  U30_M    35
## 7   myspace  U30_M     2
## 8  snapchat  U30_M     5
## 9   twitter  U30_M    23
## 10   tiktok  U30_M    60
## 11 facebook  O30.F    66
## 12  myspace  O30.F     3
## 13 snapchat  O30.F     3
## 14  twitter  O30.F    12
## 15   tiktok  O30.F     2
## 16 facebook  O30.M    58
## 17  myspace  O30.M     6
## 18 snapchat  O30.M     2
## 19  twitter  O30.M    28
## 20   tiktok  O30.M     7
```

Step 3:Use separate()/pivot_wider()

```
site_tidy <- site_tidy %>%
  separate(Gender,c("AgeGroup","Gender")) %>%
  arrange(Site)
site_tidy
```

```
##        Site AgeGroup Gender Count
## 1  facebook      U30      F    30
## 2  facebook      U30      M    35
## 3  facebook      O30      F    66
## 4  facebook      O30      M    58
## 5   myspace      U30      F     1
## 6   myspace      U30      M     2
## 7   myspace      O30      F     3
## 8   myspace      O30      M     6
## 9  snapchat      U30      F     6
## 10 snapchat      U30      M     5
## 11 snapchat      O30      F     3
## 12 snapchat      O30      M     2
## 13   tiktok      U30      F    44
## 14   tiktok      U30      M    60
## 15   tiktok      O30      F     2
## 16   tiktok      O30      M     7
## 17  twitter      U30      F    18
## 18  twitter      U30      M    23
## 19  twitter      O30      F    12
## 20  twitter      O30      M    28
```

```
site_tidy$AgeGroup <- as.factor(site_tidy$AgeGroup)
site_tidy$Gender <- as.factor(site_tidy$Gender)
site_tidy
```

```
##        Site AgeGroup Gender Count
## 1  facebook      U30      F    30
## 2  facebook      U30      M    35
## 3  facebook      O30      F    66
## 4  facebook      O30      M    58
## 5   myspace      U30      F     1
## 6   myspace      U30      M     2
## 7   myspace      O30      F     3
## 8   myspace      O30      M     6
## 9  snapchat      U30      F     6
## 10 snapchat      U30      M     5
## 11 snapchat      O30      F     3
## 12 snapchat      O30      M     2
## 13   tiktok      U30      F    44
## 14   tiktok      U30      M    60
## 15   tiktok      O30      F     2
## 16   tiktok      O30      M     7
## 17  twitter      U30      F    18
## 18  twitter      U30      M    23
## 19  twitter      O30      F    12
## 20  twitter      O30      M    28
```

Therefore, data is tidied.