Assignment 3: Data Transformation and Tidying

Krishu Thapa

2022-09-18

Question 1 (WNBA Data Sets)

Setting up the data and printing the first few value of the columns with a header containing the string ("FG").

```
suppressMessages(library(dplyr))
library(tidyr)

wnba = read.csv('WNBA_Stats_21.csv', sep=',', header = TRUE)

wnba %>% select(contains('FG')) %>% head(20)
```

```
##    FGM FGA
## 1  207 466
## 2    6  26
## 3   56 174
## 4   61 143
## 5    8  34
## 6  121 270
## 7  106 242
## 8    4  15
## 9  106 248
## 10   0   0
## 11  26  95
## 12 161 396
## 13 199 520
## 14  46 116
## 15  12  39
## 16  90 180
## 17  26  53
## 18  32  64
## 19  14  29
## 20 203 450
```

Question 1a.

The number of players with Free Throws Made > 50 and Assists > 75 are:

```
  wnba %>% na.omit() %>% filter(FTM >50 , AST > 75) %>% count()
```

```
##    n
```

```
## 1 18
```

Question 1b.

The PLAYER, TEAM, FGM, TO and PTS of players with 10 highest points in descending order of points is given by:

```
  highestpoints <-wnba %>% na.omit() %>% select(PLAYER, TEAM, FGM, TO , PTS) %>%
  arrange(desc(PTS)) %>% top_n(10)
```

```
## Selecting by PTS
```

```
  highestpoints
```

```
##                   PLAYER TEAM FGM  TO PTS
## 1          Tina Charles  WAS 238  59 631
## 2       Brittney Griner  PHO 248  66 615
## 3      Arike Ogunbowale  DAL 199  68 599
## 4           A'ja Wilson  LVA 207  46 584
## 5       Breanna Stewart  SEA 194  47 569
## 6       Kelsey Mitchell  IND 212  65 569
## 7  Skylar Diggins-Smith  PHO 177  82 566
## 8           Jewell Loyd  SEA 193  71 555
## 9        Betnijah Laney  NYL 203 119 536
## 10    Courtney Williams  ATL 228  58 529
```

```
  # Player with second highest point
  highestpoints %>% slice(2) %>% select(PLAYER)
```

```
##            PLAYER
## 1 Brittney Griner
```

Question 1c.

Adding columns FGP and FTP to the data frame and doing additional operations is shown below:

```
wnba <- wnba %>% mutate(FGP =round((FGM/FGA) * 100,2) ,
                        FTP = round((FTM/FTA) *100,2))

wnba %>% select(PLAYER, FGM, FGA , FGP , FTM, FTA, FTP) %>% head(20)
```

```
##                  PLAYER FGM FGA   FGP FTM FTA   FTP
## 1          A'ja Wilson 207 466 44.42 169 193 87.56
## 2       Aaliyah Wilson   6  26 23.08   2   4 50.00
## 3        Aari McDonald  56 174 32.18  45  51 88.24
## 4        Aerial Powers  61 143 42.66  55  60 91.67
## 5         Alanna Smith   8  34 23.53   1   4 25.00
## 6        Allie Quigley 121 270 44.81  47  49 95.92
## 7         Allisha Gray 106 242 43.80  56  65 86.15
## 8        Alyssa Thomas   4  15 26.67   3   4 75.00
## 9       Amanda Zahui B 106 248 42.74  33  43 76.74
## 10    Angel McCoughtry   0   0   NaN   0   0   NaN
## 11    Arella Guirantes  26  95 27.37  21  26 80.77
## 12        Ariel Atkins 161 396 40.66  98 118 83.05
```

```
## 13    Arike Ogunbowale 199 520 38.27 121 140 86.43
## 14    Astou Ndour-Fall  46 116 39.66  32  34 94.12
## 15          Awak Kuier  12  39 30.77  11  14 78.57
## 16       Azurá Stevens  90 180 50.00  26  32 81.25
## 17 Beatrice Mompremier  26  53 49.06   5  12 41.67
## 18        Bella Alarie  32  64 50.00  17  20 85.00
## 19     Bernadett Hatar  14  29 48.28   6   7 85.71
## 20      Betnijah Laney 203 450 45.11  96 122 78.69
```

```
wnba %>%filter(.,PLAYER=='Tina Charles') %>% select(PLAYER,FGP,FTP)
```

```
##         PLAYER   FGP   FTP
## 1 Tina Charles 44.91 82.03
```

Another two columns that we can use similarly are 3PM(3 Point Fields Goals Made) and 3PA(3 Point Field Goals Attempted), this gives us the ratio of goals made to the goals attempted for a given player and can be converted to the percentage to see the success rate of the players attempt to hit a goal. It is given by,

```
wnba <- wnba %>% mutate(X3PP = round((X3PM/X3PA) * 100,2)) %>%
  mutate_at(c("X3PP"), ~replace_na(., 0))

wnba %>% select(PLAYER, X3PM, X3PA,X3PP) %>%head(20)
```

```
##                 PLAYER X3PM X3PA   X3PP
## 1          A'ja Wilson    1    1 100.00
## 2       Aaliyah Wilson    1    7  14.29
## 3        Aari McDonald   32  104  30.77
## 4        Aerial Powers   11   35  31.43
## 5         Alanna Smith    4   21  19.05
## 6        Allie Quigley   54  119  45.38
## 7         Allisha Gray   30   82  36.59
## 8        Alyssa Thomas    0    0   0.00
## 9       Amanda Zahui B   30  107  28.04
## 10    Angel McCoughtry    0    0   0.00
## 11    Arella Guirantes    6   27  22.22
## 12        Ariel Atkins   66  184  35.87
## 13    Arike Ogunbowale   80  213  37.56
## 14    Astou Ndour-Fall    8   34  23.53
## 15          Awak Kuier    3   18  16.67
## 16       Azurá Stevens   15   45  33.33
## 17 Beatrice Mompremier    0    0   0.00
## 18        Bella Alarie    0    2   0.00
## 19     Bernadett Hatar    0    0   0.00
## 20      Betnijah Laney   34  109  31.19
```

Question 1d.

The average, min and max REB for each team in descending order of the team average is given by:

```
teamREB <- wnba %>% group_by(TEAM) %>% summarise(avgREB =
              round(mean(REB,na.rm = TRUE),2), minREB = min(REB, na.rm= TRUE),
```

```
              maxREB= max(REB, na.rm=TRUE)) %>% arrange(desc(avgREB))

teamREB
```

```
## # A tibble: 12 x 4
##    TEAM  avgREB minREB maxREB
##    <chr>  <dbl>  <int>  <int>
##  1 LVA    115.       0    298
##  2 CON    105       10    303
##  3 PHO    104.       4    302
##  4 CHI     98.9     11    193
##  5 DAL     95.8      3    173
##  6 SEA     93.9     19    267
##  7 NYL     93.6     21    171
##  8 MIN     93.3      4    312
##  9 ATL     89.5     14    219
## 10 WAS     86.6     13    258
## 11 IND     84.4      6    308
## 12 LAS     78        2    154
```

```
teamREB %>% filter(maxREB == max(maxREB)) %>% select(TEAM)
```

```
## # A tibble: 1 x 1
##   TEAM
##   <chr>
## 1 MIN
```

Question 1e.

Imputing the value for FTP and FGP.

```
wnba %>% group_by(TEAM) %>% mutate(FTP = replace_na(round((FGP/100) *
    mean(FTP,na.rm= TRUE),2))) %>% select(FTP, FGP) %>% head(20)
```

```
## Adding missing grouping variables: `TEAM`
```

```
## # A tibble: 20 x 3
## # Groups:   TEAM [11]
##    TEAM    FTP   FGP
##    <chr> <dbl> <dbl>
##  1 LVA    35.2  44.4
##  2 IND    17.0  23.1
##  3 ATL    22.7  32.2
##  4 MIN    34.5  42.7
##  5 PHO    16.8  23.5
##  6 CHI    38.7  44.8
##  7 DAL    34.9  43.8
##  8 CON    19.9  26.7
##  9 LAS    30.7  42.7
## 10 LVA    NA   NaN
## 11 LAS    19.6  27.4
## 12 WAS    32.5  40.7
## 13 DAL    30.5  38.3
## 14 CHI    34.2  39.7
```

```
## 15 DAL    24.6  30.8
## 16 CHI    43.1  50
## 17 CON    36.6  49.1
## 18 DAL    39.9  50
## 19 IND    35.7  48.3
## 20 NYL    35.4  45.1
```

```
# Second Copy

copyWNBA <- wnba

copyWNBA %>% group_by(TEAM) %>% mutate(FTP = replace_na(round(mean(FTP,na.rm= TRUE),2))) %>% select(FGP
```

```
## Adding missing grouping variables: `TEAM`
```

```
## # A tibble: 20 x 3
## # Groups:   TEAM [11]
##    TEAM    FGP   FTP
##    <chr> <dbl> <dbl>
##  1 LVA    44.4  79.3
##  2 IND    23.1  73.9
##  3 ATL    32.2  70.5
##  4 MIN    42.7  80.9
##  5 PHO    23.5  71.4
##  6 CHI    44.8  86.3
##  7 DAL    43.8  79.8
##  8 CON    26.7  74.6
##  9 LAS    42.7  71.8
## 10 LVA   NaN    79.3
## 11 LAS    27.4  71.8
## 12 WAS    40.7  79.9
## 13 DAL    38.3  79.8
## 14 CHI    39.7  86.3
## 15 DAL    30.8  79.8
## 16 CHI    50    86.3
## 17 CON    49.1  74.6
## 18 DAL    50    79.8
## 19 IND    48.3  73.9
## 20 NYL    45.1  78.4
```

In th first approach , when we multiply the FTP with the FGP ,we make an assumption that all the corresponding value of FGP exists. However this is not true as sometimes the value of FGP can also be NA which will ultimately give the value of FTP as NA too for that row. Also, the FTP value should not be dependent on the value of FGP as those are different measurement factors. In the second approach, we impute value of FTP by the average of its team and this approach is better as it is a better approach to take the mean of a particular team to decide its missing value instead of making use of mean coming from all the teams. In this approach, we will no be getting any NA value as it is dependent on single column whose NA values can be discarded while finding the mean. There are many other columns in the data whose row data contains NA and here to we can replace the NA with the average value by grouping the data as per the team. In doing so, the imputed value will not be divergent from the value that should be presented in the NA value and dosent change the overall mean by the considerable amount. This is given by,

```
wnba %>% group_by(TEAM) %>%
  mutate_if(is.numeric, ~replace_na(.,floor(mean(., na.rm = TRUE))))%>% head(20)
```

```
## `mutate_if()` ignored the following grouping variables:
## * Column `TEAM`
```

```
## # A tibble: 20 x 24
## # Groups:   TEAM [11]
##    PLAYER      TEAM    AGE     G   MIN   FGM   FGA  X3PM  X3PA   FTM   FTA  OREB
##    <chr>       <chr> <int> <int> <int> <int> <int> <int> <int> <int> <int> <int>
##  1 A'ja Wilson LVA      25    32  1021   207   466     1     1   169   193    63
##  2 Aaliyah Wi~ IND      23    14   119     6    26     1     7     2     4     5
##  3 Aari McDon~ ATL      23    30   493    56   174    32   104    45    51     9
##  4 Aerial Pow~ MIN      28    14   309    61   143    11    35    55    60    13
##  5 Alanna Smi~ PHO      25    18   117     8    34     4    21     1     4     5
##  6 Allie Quig~ CHI      36    26   635   121   270    54   119    47    49    17
##  7 Allisha Gr~ DAL      27    25   694   106   242    30    82    56    65    33
##  8 Alyssa Tho~ CON      30     2    35     4    15     0     0     3     4     1
##  9 Amanda Zah~ LAS      28    30   714   106   248    30   107    33    43    24
## 10 Angel McCo~ LVA      35     1     0     0     0     0     0     0     0     0
## 11 Arella Gui~ LAS      24    25   291    26    95     6    27    21    26     6
## 12 Ariel Atki~ WAS      25    30   918   161   396    66   184    98   118    34
## 13 Arike Ogun~ DAL      25    32  1003   199   520    80   213   121   140    23
## 14 Astou Ndou~ CHI      27    20   342    46   116     8    34    32    34    32
## 15 Awak Kuier  DAL      20    16   142    12    39     3    18    11    14     9
## 16 Azurá Stev~ CHI      26    30   587    90   180    15    45    26    32    49
## 17 Beatrice M~ CON      25    32   275    26    53     0     0     5    12    28
## 18 Bella Alar~ DAL      24    31   407    32    64     0     2    17    20    42
## 19 Bernadett ~ IND      27     7   106    14    29     0     0     6     7     7
## 20 Betnijah L~ NYL      28    32  1079   203   450    34   109    96   122    27
## # ... with 12 more variables: DREB <int>, REB <int>, AST <int>, STL <int>,
## #   BLK <int>, TO <int>, PTS <int>, DD2 <int>, TD3 <int>, FGP <dbl>, FTP <dbl>,
## #   X3PP <dbl>
```

Question 2 (Working on tidyr package for who)

Reading and setting up the data:/

```
suppressMessages(library(tidyverse))

who <- read.csv('who.csv', sep=',', header = TRUE)
```

Question 2a.

> mutate(key = stringr::str_replace(key, "newrel", "new_rel"))

The line above is necessary beacuse **rel** is a type of TB and it will be easier to seperate the rel when we place it in the form of **new_rel**. All other types of TB are extracted from the column whose names(eg: new_sp_m014 , new_sn_f65,etc) are seperated with underscore('_') and it will be consistent to represent **newrel** similarly or else when we try to seperate the type from column it will be giving the error info.

Question 2b.

The numbers of entries removed from dataset when we set values_drop_na to true in pivot longer is given by:

```
longWho <- who %>%
  pivot_longer(
    cols = new_sp_m014:newrel_f65,
    names_to = "key",
    values_to = "cases",
    values_drop_na = TRUE
  )


# Count of the dropped data sets when na values are dropped

wideWho <- longWho %>%
pivot_wider(names_from = key, values_from = cases)

count(who) - count(wideWho)
```

```
##      n
## 1 3756
```

Question 2c.

The difference between an explicit and implicit missing value is that in explicit case the missing value is clearly represented by the *NA* however, in the implicit missing value there is no clear inidication of the missing value in the dataset. When the pivot_longer version of dataset is widened on the column "Key" with the values from column "cases", we can see multiple of the columns derived from the "Key" column to hold the NA value. Here, the implicit value is explicitly presented this way.

```
longWho <- who %>%
  pivot_longer(
    cols = new_sp_m014:newrel_f65,
    names_to = "key",
```

```
    values_to = "cases",
    values_drop_na = TRUE
  )

wideWho <- longWho %>%
pivot_wider(names_from = key, values_from = cases)

wideWho %>% head(20)
```

```
## # A tibble: 20 x 60
##    country     iso2  iso3   year new_s~1 new_s~2 new_s~3 new_s~4 new_s~5 new_s~6
##    <chr>       <chr> <chr> <int>   <int>   <int>   <int>   <int>   <int>   <int>
##  1 Afghanistan AF    AFG    1997       0      10       6       3       5       2
##  2 Afghanistan AF    AFG    1998      30     129     128      90      89      64
##  3 Afghanistan AF    AFG    1999       8      55      55      47      34      21
##  4 Afghanistan AF    AFG    2000      52     228     183     149     129      94
##  5 Afghanistan AF    AFG    2001     129     379     349     274     204     139
##  6 Afghanistan AF    AFG    2002      90     476     481     368     246     241
##  7 Afghanistan AF    AFG    2003     127     511     436     284     256     288
##  8 Afghanistan AF    AFG    2004     139     537     568     360     358     386
##  9 Afghanistan AF    AFG    2005     151     606     560     472     453     470
## 10 Afghanistan AF    AFG    2006     193     837     791     574     572     572
## 11 Afghanistan AF    AFG    2007     186     856     840     597     566     630
## 12 Afghanistan AF    AFG    2008     187     941     773     545     570     630
## 13 Afghanistan AF    AFG    2009     200     906     705     499     491     596
## 14 Afghanistan AF    AFG    2010     197     986     819     491     490     641
## 15 Afghanistan AF    AFG    2011     204    1010     895     613     570     700
## 16 Afghanistan AF    AFG    2012     188    1116     801     586     521     585
## 17 Afghanistan AF    AFG    2013      NA      NA      NA      NA      NA      NA
## 18 Albania     AL    ALB    1995       0       0       0       0      19      40
## 19 Albania     AL    ALB    1997       0      23      43      33      25      21
## 20 Albania     AL    ALB    1998       1      17      21      24      18      26
## # ... with 50 more variables: 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_ep_m014 <int>, new_ep_m1524 <int>,
## #   new_ep_m2534 <int>, new_ep_m3544 <int>, new_ep_m4554 <int>, ...
```

Question 2d.

Looking at the complete data and intepreting the type is given below:

```
tidyData <- 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)

tidyData
```

```
## # 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
```

```
str(tidyData)
```

```
## tibble [76,046 x 6] (S3: tbl_df/tbl/data.frame)
##  $ country: chr [1:76046] "Afghanistan" "Afghanistan" "Afghanistan" "Afghanistan" ...
##  $ year   : int [1:76046] 1997 1997 1997 1997 1997 1997 1997 1997 1997 1997 ...
##  $ var    : chr [1:76046] "sp" "sp" "sp" "sp" ...
##  $ sex    : chr [1:76046] "m" "m" "m" "m" ...
##  $ age    : chr [1:76046] "014" "1524" "2534" "3544" ...
##  $ cases  : int [1:76046] 0 10 6 3 5 2 0 5 38 36 ...
```

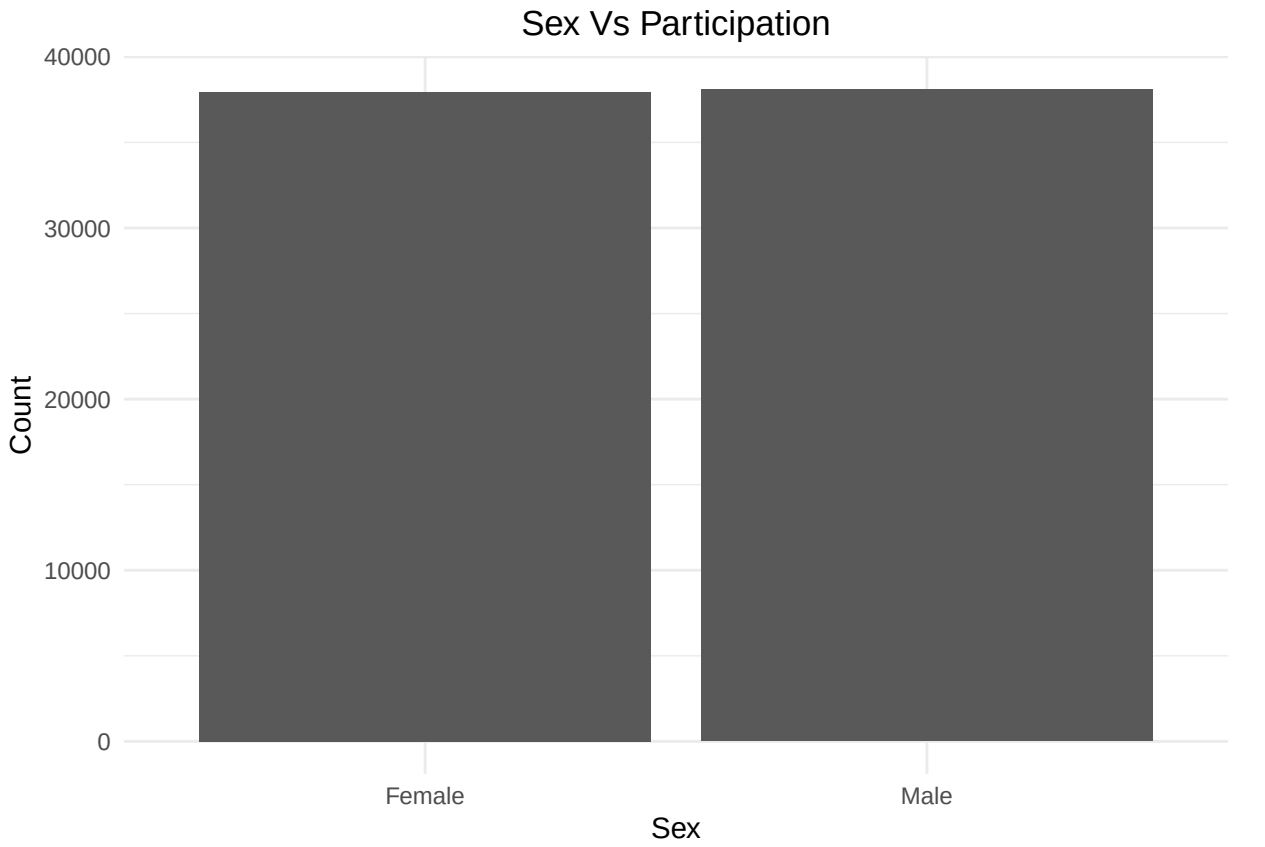
I think since the countries, year, var, sex and age represent the data corresponding to different categories and levels. I think these fields should be represented in the form of **factor** rather than presenting them as the **char**.

Question 2e.

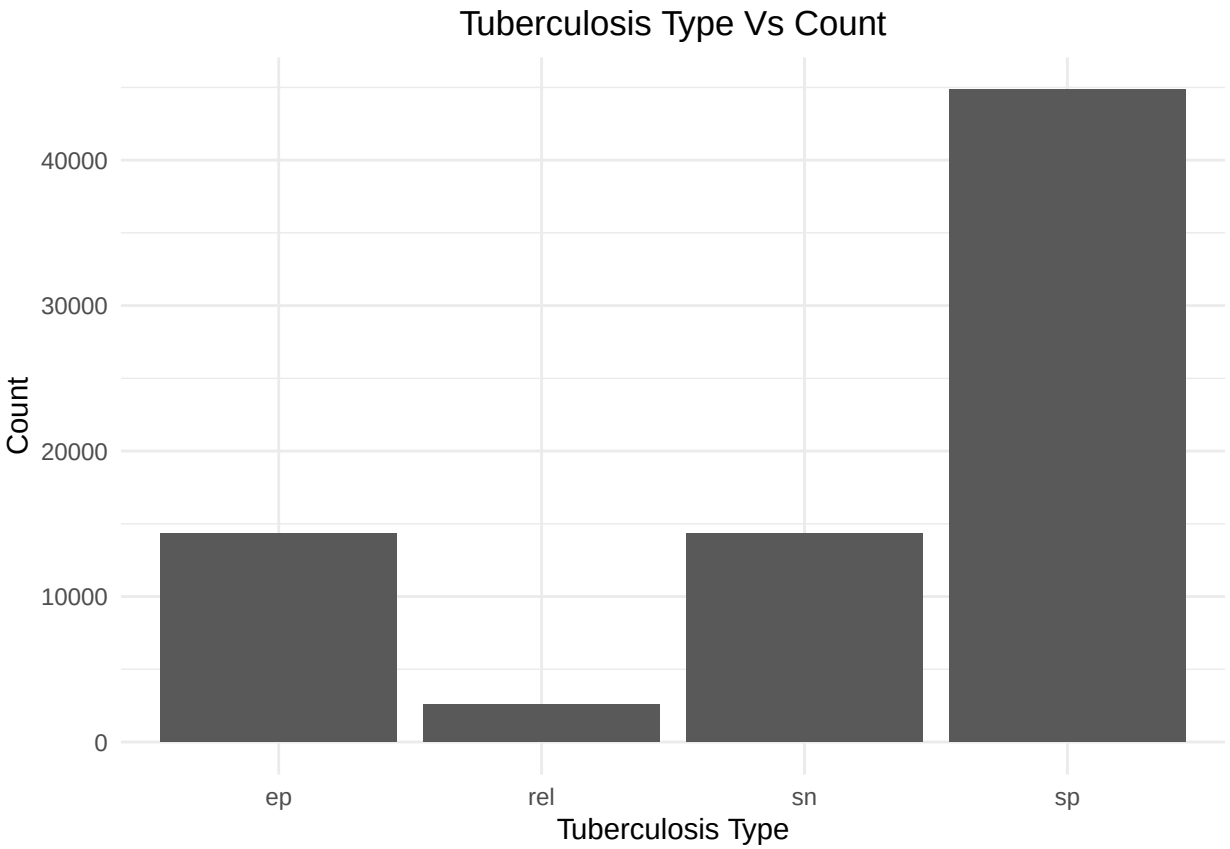
Visualization for the data is given by,

```
library(ggplot2)


## Visualtization 1 (Sex Participation)

maleCount <- tidyData %>% filter(sex=='m') %>% nrow()
femaleCount <- tidyData %>% filter(sex=='f') %>% nrow()

sexPlot <- tibble(sex= c("Male", "Female"), count= c(maleCount, femaleCount))

ggplot(data=sexPlot, aes(x=sex, y= count)) +
  geom_bar(stat="identity")+theme_minimal()+
```

```
  labs(x="Sex", y="Count", title="Sex Vs Participation")+
  theme(plot.title = element_text(hjust = 0.5))
```




```
# Visualization 2 (Tuberculosis type distribution)

varPlot <- tidyData %>% group_by(var) %>% count()

ggplot(data=varPlot, aes(x=var, y = n)) +
  geom_bar(stat="identity")+theme_minimal()+
  labs(x="Tuberculosis Type", y="Count", title="Tuberculosis Type Vs Count")+
  theme(plot.title = element_text(hjust = 0.5))
```



From the first visualization we can see that almost same number of male and female were surveyed for the data.Similarly, from the second visualization we can see that the tuberculosis type(sp) has more number of cases as compared to other types. I thought it would be interesting to check over the sex because we can identify how common are tuberculosis between male and female. Also , I thought it would be interesting to test over tuberculosis type so that we can identify which type of tuberculosis is more prevalent in the tuberculosis patients.

Question 2f.

The implementation for the given ques is given below:

```
# Reading the data

schQtr = read.csv('SchQtr.csv', sep=',', header = TRUE)

tidySchQtr <- schQtr %>% pivot_longer(cols= Qtr.1:Qtr.4, names_to= "Quarter" ,
          values_to= "Student_Count", values_drop_na = TRUE) %>% mutate(
    Quarter = stringr::str_replace(Quarter, "Qtr_2", "Qtr.2"))%>%
  separate(Quarter, c("Interval_Type","Interval_Id"))

tidySchQtr %>% head(10)
```

```
## # A tibble: 10 x 5
##    School  Year Interval_Type Interval_Id Student_Count
##    <chr>  <int> <chr>         <chr>               <int>
##  1 UNI     2018 Qtr           1                      27
##  2 UNI     2018 Qtr           2                      90
```

```
##  3 UNI     2018 Qtr           3                     12
##  4 UNI     2018 Qtr           4                     84
##  5 COL     2018 Qtr           1                     42
##  6 COL     2018 Qtr           2                     27
##  7 COL     2018 Qtr           3                     62
##  8 COL     2018 Qtr           4                      1
##  9 ACA     2018 Qtr           1                      6
## 10 ACA     2018 Qtr           2                     51
```

```
# Count of the new restructured table.
nrow(tidySchQtr)
```

```
## [1] 48
```