Data Analysis and Visualization Exercise 3, Data Wrangling

Vicente Yépez, Žiga Avsec

5 November 2019

Setup

```
library(data.table)
library(magrittr)   # Needed for %>% operator
library(tidyr)
library(readxl)
library(dplyr)
DATADIR <- "../extdata"
```

Questions

Section I. Reading and cleaning up data

1. Read the pokemon.xlsx file. Assign the contents of each of the four sheets (tabs) into different data.tables: poke_dt, moves_dt, evolution_dt, typeChart_dt. Open the data.tables and understand the data they contain. *Hint:* read_excel from the readxl package.

```
poke_file <- file.path(DATADIR, 'pokemon.xlsx')
poke_dt <- read_excel(poke_file, sheet="Pokemon")
class(poke_dt)  # read_excel doesn't return a data.table, so we need to convert it to one
## [1] "tbl_df"     "tbl"        "data.frame"
poke_dt <- as.data.table(poke_dt)
head(poke_dt)
##       #      Name   Type Total HP Attack Defense Special Attack
## 1:  001 Bulbasaur  GRASS   318 45     49      49             65
## 2:  001 Bulbasaur POISON   318 45     49      49             65
## 3:  002   Ivysaur  GRASS   405 60     62      63             80
## 4:  002   Ivysaur POISON   405 60     62      63             80
## 5:  003  Venusaur  GRASS   525 80     82      83            100
## 6:  003  Venusaur POISON   525 80     82      83            100
##    Special Defense Speed
## 1:              65    45
## 2:              65    45
## 3:              80    60
## 4:              80    60
```

```
## 5:             100    80
## 6:             100    80

moves_dt <- read_excel(poke_file, sheet = "Moves") %>% as.data.table
evolution_dt <- read_excel(poke_file, sheet = "Evolution") %>% as.data.table
typeChart_dt <- read_excel(poke_file, sheet = "TypeChart") %>% as.data.table
```

2. Check the columns' names and classes of poke_dt. Rename the column "#" as "Number". Replace the spaces in the columns' names with underscores. *Hint:* colnames(), setnames(), gsub()

```
colnames(poke_dt)
##  [1] "#"               "Name"            "Type"            "Total"
##  [5] "HP"              "Attack"          "Defense"         "Special Attack"
##  [9] "Special Defense" "Speed"
setnames(poke_dt, "#", "Number")
# gsub example
gsub(pattern = "cup", replacement = "phone", x = c("this is a nice cup", "two cups"))
## [1] "this is a nice phone" "two phones"

colnames(poke_dt) <- gsub(" ", "_", colnames(poke_dt))
sapply(poke_dt, class)
##          Number            Name            Type           Total
##     "character"     "character"     "character"       "numeric"
##              HP          Attack         Defense  Special_Attack
##       "numeric"       "numeric"       "numeric"       "numeric"
## Special_Defense           Speed
##       "numeric"       "numeric"
```

3. Note that the "Number" column is a character vector that sometimes begins with a space. To remove this space use the following:

```
# poke_dt[, Number := gsub(intToUtf8(160),'', Number)]
```

Then, change the class of "Number" from character to integer and "Type" from character to factor.

Hint: as.integer(), as.factor()

```
# intToUtf8(160) -> weird character (?)
intToUtf8(160) == "_" # FALSE
## [1] FALSE
intToUtf8(160) == " " # FALSE
## [1] FALSE
poke_dt[, Number := gsub(intToUtf8(160),'', Number)]
poke_dt[, Number := as.integer(Number)]
poke_dt[, Type := as.factor(Type)]
sapply(poke_dt, class)
##          Number            Name            Type           Total
##       "integer"     "character"        "factor"       "numeric"
##              HP          Attack         Defense  Special_Attack
##       "numeric"       "numeric"       "numeric"       "numeric"
```

```
## Special_Defense           Speed
##       "numeric"       "numeric"
```

4. Remove all 'Mega' pokemons. Then, subset to include pokemon from the first generation only (from Bulbasaur #1 until Mewtwo #150). *Hint:* grep()

```
grep("Mega ", poke_dt$Name, value = T)
##  [1] "Mega Venusaur"    "Mega Venusaur"    "Mega Charizard X"
##  [4] "Mega Charizard X" "Mega Charizard Y" "Mega Charizard Y"
##  [7] "Mega Blastoise"   "Mega Blastoise"   "Mega Alakazam"
## [10] "Mega Gengar"      "Mega Gengar"      "Mega Kangaskhan"
## [13] "Mega Kangaskhan"  "Mega Pinsir"      "Mega Pinsir"
## [16] "Mega Gyarados"    "Mega Gyarados"    "Mega Aerodactyl"
## [19] "Mega Aerodactyl"  "Mega Mewtwo X"    "Mega Mewtwo X"
## [22] "Mega Mewtwo Y"    "Mega Mewtwo Y"    "Mega Ampharos"
## [25] "Mega Ampharos"    "Mega Scizor"      "Mega Scizor"
## [28] "Mega Heracross"   "Mega Heracross"   "Mega Houndoom"
## [31] "Mega Houndoom"    "Mega Tyranitar"   "Mega Tyranitar"
## [34] "Mega Blaziken"    "Mega Blaziken"    "Mega Gardevoir"
## [37] "Mega Gardevoir"   "Mega Mawile"      "Mega Mawile"
## [40] "Mega Aggron"      "Mega Aggron"      "Mega Medicham"
## [43] "Mega Medicham"    "Mega Manectric"   "Mega Manectric"
## [46] "Mega Banette"     "Mega Banette"     "Mega Absol"
## [49] "Mega Absol"       "Mega Garchomp"    "Mega Garchomp"
## [52] "Mega Lucario"     "Mega Lucario"     "Mega Abomasnow"
## [55] "Mega Abomasnow"
poke_dt <- poke_dt[!(grep("Mega ", Name))] # Mind the space after 'Mega'
# Also poke_dt<-poke_dt[grep("Mega ", Name, invert=TRUE)]
poke_dt <- poke_dt[Number <= 150]
```

Section II. Data exploration

1. Which are all the types of pokemons? How many pokemon are there per type?

```
unique(poke_dt$Type)
##  [1] GRASS    POISON   FIRE     FLYING   WATER    BUG      NORMAL   ELECTRIC
##  [9] GROUND   FAIRY    FIGHTING PSYCHIC  ROCK     STEEL    ICE      GHOST
## [17] DRAGON
## 18 Levels: BUG DARK DRAGON ELECTRIC FAIRY FIGHTING FIRE FLYING ... WATER
poke_dt[, .N, by = Type]   # Also: table(poke_dt$Type)
##         Type  N
##  1:    GRASS 14
##  2:   POISON 33
##  3:     FIRE 12
##  4:   FLYING 19
##  5:    WATER 32
##  6:      BUG 12
##  7:   NORMAL 22
##  8: ELECTRIC  9
##  9:   GROUND 14
```

```
## 10:    FAIRY  5
## 11: FIGHTING  8
## 12:  PSYCHIC 13
## 13:     ROCK 11
## 14:    STEEL  2
## 15:      ICE  5
## 16:    GHOST  3
## 17:   DRAGON  3
# sort the types by N
#poke_dt[, .N, by = Type][order(-N)]
# what is order() doing?
#v <- c(5, 2, 3)
#v[order(v)]
```

2. Which are all the Ice pokemon?

```
poke_dt[Type == "ICE"]
##    Number     Name Type Total  HP Attack Defense Special_Attack
## 1:     87  Dewgong  ICE   475  90     70      80             70
## 2:     91 Cloyster  ICE   525  50     95     180             85
## 3:    124     Jynx  ICE   455  65     50      35            115
## 4:    131   Lapras  ICE   535 130     85      80             85
## 5:    144 Articuno  ICE   580  90     85     100             95
##    Special_Defense Speed
## 1:              95    70
## 2:              45    70
## 3:              95    95
## 4:              95    60
## 5:             125    85
```

3. Create a character vector called stats that contains the column names of the pokemons stats (HP, attack, etc.). Which is the maximum value of each stat? First, think how would it be just for one, eg. HP. *Hint:* sapply (or .SD)

```
stats <- c("HP", "Attack", "Defense", "Special_Attack", "Special_Defense", "Speed", "Total")
```

```
stopifnot(all(stats %in% names(poke_dt))) # sanity check - all are in the table
poke_dt[, stats, with = F]  # get only the stats columns. Keep in mind the notation
##       HP Attack Defense Special_Attack Special_Defense Speed Total
##   1:  45     49      49             65              65    45   318
##   2:  45     49      49             65              65    45   318
##   3:  60     62      63             80              80    60   405
##   4:  60     62      63             80              80    60   405
##   5:  80     82      83            100             100    80   525
##  ---
## 213:  41     64      45             50              50    50   300
## 214:  61     84      65             70              70    70   420
## 215:  91    134      95            100             100    80   600
## 216:  91    134      95            100             100    80   600
## 217: 106    110      90            154              90   130   680
poke_dt[, ..stats]  # same as before, but different data.table notation
```

```
##       HP Attack Defense Special_Attack Special_Defense Speed Total
##   1:  45     49      49             65              65    45   318
##   2:  45     49      49             65              65    45   318
##   3:  60     62      63             80              80    60   405
##   4:  60     62      63             80              80    60   405
##   5:  80     82      83            100             100    80   525
##  ---
## 213:  41     64      45             50              50    50   300
## 214:  61     84      65             70              70    70   420
## 215:  91    134      95            100             100    80   600
## 216:  91    134      95            100             100    80   600
## 217: 106    110      90            154              90   130   680

# just for HP
poke_dt[, max(HP)]
## [1] 250
poke_dt[, sapply(.SD, max), .SDcols = stats]  # sapply returns a vector
##              HP          Attack         Defense  Special_Attack
##             250             134             180             154
## Special_Defense           Speed           Total
##             125             140             680
sapply(poke_dt[, stats, with = F], max)
##              HP          Attack         Defense  Special_Attack
##             250             134             180             154
## Special_Defense           Speed           Total
##             125             140             680
sapply(poke_dt[, ..stats], max)   # similar to the line above
##              HP          Attack         Defense  Special_Attack
##             250             134             180             154
## Special_Defense           Speed           Total
##             125             140             680
```

4. Remember that R has the functions `colMeans` and `colSums`. Create the function `colMax(DT)` that takes as input a data.table and returns a named vector with the maximum value of all columns. Use it to obtain the same results as in the previous exercise.

```
colMax <- function(DT){
  stopifnot(class(DT)[1] == "data.table")
  DT[, sapply(.SD, max)]
  # or sapply(DT, max)
  }
colMax(poke_dt[,..stats])
##              HP          Attack         Defense  Special_Attack
##             250             134             180             154
## Special_Defense           Speed           Total
##             125             140             680
```

5. Go to the help page of `which.min`. You will see that this function returns the index of the **first** minimum of a vector. Create the functions `which_min` and `which_max` whose input is a numeric vector and output is a vector with **all** the indices where the minimum and maximum values are located. *Hint:* which()

```
which.min(c(1,1,3))  # returns only the first occurrence
## [1] 1
which_max = function(x)  which(x == max(x, na.rm=T))   # It's important to add the na.rm parameter
which_min = function(x)  which(x == min(x, na.rm=T))
```

6. Of each Type, which are the: first, strongest and weakest pokemon? We define 'strongest' as the one with the highest Total value. *Hint*: .SD, which_max()

```
poke_dt[, .SD[1], by = Type]
##         Type Number       Name Total HP Attack Defense Special_Attack
##  1:    GRASS      1  Bulbasaur   318 45     49      49             65
##  2:   POISON      1  Bulbasaur   318 45     49      49             65
##  3:     FIRE      4 Charmander   309 39     52      43             60
##  4:   FLYING      6  Charizard   534 78     84      78            109
##  5:    WATER      7   Squirtle   314 44     48      65             50
##  6:      BUG     10   Caterpie   195 45     30      35             20
##  7:   NORMAL     16     Pidgey   251 40     45      40             35
##  8: ELECTRIC     25    Pikachu   300 35     55      30             50
##  9:   GROUND     27  Sandshrew   300 50     75      85             20
## 10:    FAIRY     35   Clefairy   323 70     45      48             60
## 11: FIGHTING     56     Mankey   305 40     80      35             35
## 12:  PSYCHIC     63       Abra   310 25     20      15            105
## 13:     ROCK     74    Geodude   300 40     80     100             30
## 14:    STEEL     81  Magnemite   325 25     35      70             95
## 15:      ICE     87    Dewgong   475 90     70      80             70
## 16:    GHOST     92     Gastly   310 30     35      30            100
## 17:   DRAGON    147    Dratini   300 41     64      45             50
##     Special_Defense Speed
##  1:              65    45
##  2:              65    45
##  3:              50    65
##  4:              85   100
##  5:              64    43
##  6:              20    45
##  7:              35    56
##  8:              40    90
##  9:              30    40
## 10:              65    35
## 11:              45    70
## 12:              55    90
## 13:              30    20
## 14:              55    45
## 15:              95    70
## 16:              35    80
## 17:              50    50
poke_dt[, .SD[which_max(Total)], by = Type]
##         Type Number       Name Total  HP Attack Defense Special_Attack
##  1:    GRASS      3   Venusaur   525  80     82      83            100
##  2:   POISON      3   Venusaur   525  80     82      83            100
##  3:     FIRE    146    Moltres   580  90    100      90            125
##  4:   FLYING    149  Dragonite   600  91    134      95            100
```

```
##  5:    WATER    130   Gyarados   540  95    125      79            60
##  6:      BUG    123    Scyther   500  70    110      80            55
##  7:      BUG    127     Pinsir   500  65    125     100            55
##  8:   NORMAL    143    Snorlax   540 160    110      65            65
##  9: ELECTRIC    145     Zapdos   580  90     90      85           125
## 10:   GROUND     31  Nidoqueen   495  90     82      87            75
## 11:   GROUND     34   Nidoking   495  81     92      77            85
## 12:    FAIRY     36   Clefable   473  95     70      73            85
## 13: FIGHTING     68    Machamp   505  90    130      80            65
## 14:  PSYCHIC    150     Mewtwo   680 106    110      90           154
## 15:     ROCK    142 Aerodactyl   515  80    105      65            60
## 16:    STEEL     82   Magneton   465  50     60      95           120
## 17:      ICE    144   Articuno   580  90     85     100            95
## 18:    GHOST     94     Gengar   500  60     65      60           130
## 19:   DRAGON    149  Dragonite   600  91    134      95           100
##     Special_Defense Speed
##  1:             100    80
##  2:             100    80
##  3:              85    90
##  4:             100    80
##  5:             100    81
##  6:              80   105
##  7:              70    85
##  8:             110    30
##  9:              90   100
## 10:              85    76
## 11:              75    85
## 12:              90    60
## 13:              85    55
## 14:              90   130
## 15:              75   130
## 16:              70    70
## 17:             125    85
## 18:              75   110
## 19:             100    80
poke_dt[, .SD[which_min(Total)], by = Type]
##         Type Number       Name Total  HP Attack Defense Special_Attack
##  1:    GRASS     46      Paras   285  35     70      55            45
##  2:   POISON     13     Weedle   195  40     35      30            20
##  3:     FIRE     37     Vulpix   299  38     41      40            50
##  4:   FLYING     41      Zubat   245  40     45      35            30
##  5:    WATER    129   Magikarp   200  20     10      55            15
##  6:      BUG     10   Caterpie   195  45     30      35            20
##  7:      BUG     13     Weedle   195  40     35      30            20
##  8:   NORMAL     16     Pidgey   251  40     45      40            35
##  9: ELECTRIC     25    Pikachu   300  35     55      30            50
## 10:   GROUND     50    Diglett   265  10     55      25            35
## 11:    FAIRY     39 Jigglypuff   270 115     45      20            45
## 12: FIGHTING     56     Mankey   305  40     80      35            35
## 13: FIGHTING     66     Machop   305  70     80      50            35
## 14:  PSYCHIC     63       Abra   310  25     20      15           105
```

```
## 15:     ROCK     74    Geodude   300  40     80     100             30
## 16:    STEEL     81  Magnemite   325  25     35      70             95
## 17:      ICE    124       Jynx   455  65     50      35            115
## 18:    GHOST     92     Gastly   310  30     35      30            100
## 19:   DRAGON    147    Dratini   300  41     64      45             50
##     Special_Defense Speed
##  1:              55    25
##  2:              20    50
##  3:              65    65
##  4:              40    55
##  5:              20    80
##  6:              20    45
##  7:              20    50
##  8:              35    56
##  9:              40    90
## 10:              45    95
## 11:              25    20
## 12:              45    70
## 13:              35    35
## 14:              55    90
## 15:              30    20
## 16:              55    45
## 17:              95    95
## 18:              35    80
## 19:              50    50
```

7. On average, which is the strongest Type? Specifically, give a sorted data.table with all pokemon types and their 'Total' means. *Hint*: order()

```
poke_dt[, .(av_Total = mean(Total)), by = Type][order(av_Total)] # Other approach
##         Type av_Total
##  1:      BUG 334.5833
##  2:   NORMAL 379.5000
##  3:   POISON 380.8182
##  4:    FAIRY 390.2000
##  5:    STEEL 395.0000
##  6:   GROUND 396.0714
##  7:    GRASS 398.7857
##  8:    GHOST 405.0000
##  9:    WATER 412.6250
## 10:     ROCK 418.6364
## 11: FIGHTING 423.1250
## 12:   DRAGON 440.0000
## 13: ELECTRIC 441.1111
## 14:   FLYING 442.5789
## 15:  PSYCHIC 444.3077
## 16:     FIRE 455.5833
## 17:      ICE 514.0000
```

Section III. Merging and manipulating

1. In evolution_dt, rename the columns 'Evolving from' -> 'Name' and 'Evolving to' -> 'Evolution'. Subset it in order to include only the pokemons from the first generation (As subsetted in question section I question 4).

```
setnames(evolution_dt, c("Evolving from", "Evolving to"), c("Name", "Evolution"))
head(evolution_dt)
##          Name  Evolution Level Condition Evolution Type
## 1:  Bulbasaur    Ivysaur    16      <NA>          Level
## 2:    Ivysaur   Venusaur    32      <NA>          Level
## 3: Charmander Charmeleon    16      <NA>          Level
## 4: Charmeleon  Charizard    36      <NA>          Level
## 5:   Squirtle  Wartortle    16      <NA>          Level
## 6:  Wartortle  Blastoise    36      <NA>          Level
evo_dt <- evolution_dt[Name %in% poke_dt$Name & Evolution %in% poke_dt$Name]
```

2. Which pokemons neither evolve nor are an evolution from others? *Hint*: There's no need to merge poke_dt with evo_dt, we are simply interested in the pokemon that are not in `evo_dt$Name` or `evo_dt$Evolution`.

```
# don't evolve, are not an evolution of others <==> not present in evolution_dt
setdiff(poke_dt$Name, c(evo_dt$Name, evo_dt$Evolution))
##  [1] "Farfetch'd" "Onix"       "Hitmonlee"  "Hitmonchan" "Lickitung"
##  [6] "Chansey"    "Tangela"    "Kangaskhan" "Mr. Mime"   "Scyther"
## [11] "Jynx"       "Electabuzz" "Magmar"     "Pinsir"     "Tauros"
## [16] "Lapras"     "Ditto"      "Porygon"    "Aerodactyl" "Snorlax"
## [21] "Articuno"   "Zapdos"     "Moltres"    "Mewtwo"
```

3. Add a column to the poke_dt with the 1st generation evolutions of each pokemon (only the first one) and the level it requires to evolve. *Hint*: merge()

```
# Using merge
poke_merge <- merge(poke_dt, evo_dt[,.(Name,Evolution,Level)], by="Name")  # Only pokemon with evolutions
poke_merge <- merge(poke_dt, evo_dt[,.(Name,Evolution,Level)], by="Name", all.x = T, sort = F)
```

4. Which pokemon requires the highest level to evolve? By type?

```
poke_merge[which_max(Level)]
##         Name Number   Type Total HP Attack Defense Special_Attack
## 1: Dragonair    148 DRAGON   420 61     84      65             70
##    Special_Defense Speed Evolution Level
## 1:              70    70 Dragonite    55
poke_merge[, .SD[which_max(Level)], by = Type]
##         Type      Name Number Total HP Attack Defense Special_Attack
##  1:    GRASS   Ivysaur      2   405 60     62      63             80
##  2:   POISON    Grimer     88   325 80     80      50             40
##  3:     FIRE    Ponyta     77   410 50     85      55             65
##  4:   FLYING Pidgeotto     17   349 63     60      55             50
##  5:    WATER   Omanyte    138   355 35     40     100             90
##  6:    WATER    Kabuto    140   355 30     80      90             55
##  7:      BUG   Venonat     48   305 60     55      50             40
```

```
##  8:   NORMAL Pidgeotto     17   349 63     60      55             50
##  9: ELECTRIC Magnemite     81   325 25     35      70             95
## 10: ELECTRIC   Voltorb    100   330 40     30      50             55
## 11:   GROUND   Rhyhorn    111   345 80     85      95             30
## 12: FIGHTING    Mankey     56   305 40     80      35             35
## 13: FIGHTING    Machop     66   305 70     80      50             35
## 14:  PSYCHIC  Slowpoke     79   315 90     65      65             40
## 15:     ROCK   Rhyhorn    111   345 80     85      95             30
## 16:    STEEL Magnemite     81   325 25     35      70             95
## 17:    GHOST    Gastly     92   310 30     35      30            100
## 18:   DRAGON Dragonair    148   420 61     84      65             70
##     Special_Defense Speed Evolution Level
##  1:              80    60  Venusaur    32
##  2:              50    25       Muk    38
##  3:              65    90  Rapidash    40
##  4:              50    71   Pidgeot    36
##  5:              55    35   Omastar    40
##  6:              45    55  Kabutops    40
##  7:              55    45  Venomoth    31
##  8:              50    71   Pidgeot    36
##  9:              55    45  Magneton    30
## 10:              55   100 Electrode    30
## 11:              30    25    Rhydon    42
## 12:              45    70  Primeape    28
## 13:              35    35   Machoke    28
## 14:              40    15   Slowbro    37
## 15:              30    25    Rhydon    42
## 16:              55    45  Magneton    30
## 17:              35    80   Haunter    25
## 18:              70    70 Dragonite    55
```

5. Now let's examine `moves_dt`. It contains all moves (or attacks) that pokemons can do. Check the columns' classes. Try to explain the first row.

```
head(moves_dt)
##           Name   Type     Cat. Power Acc. PP   TM
## 1:      Absorb  GRASS  Special    20  100 25 <NA>
## 2:        Acid POISON  Special    40  100 30 <NA>
## 3:  Acid Armor POISON   Status    NA <NA> 40 <NA>
## 4:  Acid Spray POISON  Special    40  100 20 <NA>
## 5:  Acrobatics FLYING Physical    55  100 15 TM62
## 6: Acupressure NORMAL   Status    NA <NA> 30 <NA>
##                                              Effect Prob. (%)
## 1:  User recovers half the HP inflicted on opponent.        NA
## 2:             May lower opponent's Special Defense.        10
## 3:                   Sharply raises user's Defense.        NA
## 4:        Sharply lowers opponent's Special Defense.       100
## 5: Stronger when the user does not have a held item.        NA
## 6:                    Sharply raises a random stat.        NA
colnames(moves_dt)
## [1] "Name"      "Type"      "Cat."      "Power"     "Acc."      "PP"
```

```
## [7] "TM"        "Effect"    "Prob. (%)"
sapply(moves_dt, class)
##        Name        Type        Cat.       Power        Acc.          PP
## "character" "character" "character"   "numeric" "character"   "numeric"
##          TM      Effect   Prob. (%)
## "character" "character"   "numeric"
```

6. Expand the names (use the full names) of the columns Cat., Acc. and Prob. Change Accuracy class to numeric.

```
setnames(moves_dt, c("Cat.","Acc.", "Prob. (%)"), c("Category","Accuracy", "Probability"))
moves_dt[, Accuracy := as.numeric(Accuracy)]
```

7. Which are the different categories of the moves?

```
unique(moves_dt$Category)
## [1] "Special"  "Status"   "Physical"
```

8. Which are all the Grass attacks?

```
moves_dt[Type == "GRASS",] %>% head
##            Name  Type Category Power Accuracy PP   TM
## 1:       Absorb GRASS  Special    20      100 25 <NA>
## 2: Aromatherapy GRASS   Status    NA       NA  5 <NA>
## 3:  Bullet Seed GRASS Physical    25      100 30 <NA>
## 4: Cotton Guard GRASS   Status    NA       NA 10 <NA>
## 5: Cotton Spore GRASS   Status    NA      100 40 <NA>
## 6:  Energy Ball GRASS  Special    80      100 10 TM53
##                                            Effect Probability
## 1: User recovers half the HP inflicted on opponent.          NA
## 2:         Cures all status problems in your party.          NA
## 3:                      Hits 25 times in one turn.          NA
## 4:               Drastically raises user's Defense.          NA
## 5:                 Sharply lowers opponent's Speed.          NA
## 6:           May lower opponent's Special Defense.          10
```

9. Assuming that a pokemon can only perform attacks based on its type(s), which are all the attacks that eg. Bulbasaur can do? There is no need to merge poke_dt with moves_dt.

```
moves_dt[Type %in% poke_dt[Name == "Bulbasaur", Type]] %>% head
##            Name   Type Category Power Accuracy PP   TM
## 1:       Absorb  GRASS  Special    20      100 25 <NA>
## 2:         Acid POISON  Special    40      100 30 <NA>
## 3:   Acid Armor POISON   Status    NA       NA 40 <NA>
## 4:   Acid Spray POISON  Special    40      100 20 <NA>
## 5: Aromatherapy  GRASS   Status    NA       NA  5 <NA>
## 6:        Belch POISON  Special   120       90 10 <NA>
##                                          Effect Probability
## 1: User recovers half the HP inflicted on opponent.          NA
## 2:           May lower opponent's Special Defense.          10
## 3:                  Sharply raises user's Defense.          NA
```

```
## 4:       Sharply lowers opponent's Special Defense.         100
## 5:         Cures all status problems in your party.          NA
## 6:                 Requires a held Berry to attack.          NA
```

10. Add a column `Real_Power` with the real power of an attack which we define as Power * Accuracy %. Which is the most powerful attack of all? Which are the most powerful attacks per type and category? *Hint:* by = .(Category, Type)

```
moves_dt[, Real_Power := Power * Accuracy / 100]
moves_dt[which_max(Real_Power)]
##         Name   Type Category Power Accuracy PP   TM       Effect Probability
## 1: Explosion NORMAL Physical   250      100  5 TM64 User faints.          NA
##    Real_Power
## 1:        250
moves_dt[, .SD[which_max(Real_Power)], by = Type] %>% head
##       Type         Name Category Power Accuracy PP   TM
## 1:   GRASS Frenzy Plant  Special   150       90  5 <NA>
## 2:  POISON        Belch  Special   120       90 10 <NA>
## 3:  FLYING   Sky Attack Physical   140       90  5 <NA>
## 4:  NORMAL    Explosion Physical   250      100  5 TM64
## 5: PSYCHIC Psycho Boost  Special   140       90  5 <NA>
## 6:    ROCK Rock Wrecker Physical   150       90  5 <NA>
##                                                        Effect
## 1:                                User must recharge next turn.
## 2:                              Requires a held Berry to attack.
## 3: Charges on first turn, attacks on second. May cause flinching.
## 4:                                                  User faints.
## 5:                           Sharply lowers user's Special Attack.
## 6:                                  User must recharge next turn.
##    Probability Real_Power
## 1:          NA        135
## 2:          NA        108
## 3:          30        126
## 4:          NA        250
## 5:          NA        126
## 6:          NA        135
moves_dt[, .SD[which_max(Real_Power)], by = .(Category,Type)] %>% head
##    Category   Type         Name Power Accuracy PP   TM
## 1:  Special  GRASS Frenzy Plant   150       90  5 <NA>
## 2:  Special POISON        Belch   120       90 10 <NA>
## 3: Physical FLYING   Sky Attack   140       90  5 <NA>
## 4:  Special FLYING    Aeroblast   100       95  5 <NA>
## 5:  Special   ROCK    Power Gem    70      100 20 <NA>
## 6: Physical  WATER    Aqua Tail    90       90 10 <NA>
##                                                      Effect
## 1:                                User must recharge next turn.
## 2:                              Requires a held Berry to attack.
## 3: Charges on first turn, attacks on second. May cause flinching.
## 4:                                       High critical hit ratio.
## 5:                                                         <NA>
## 6:                                                         <NA>
```

```
##    Probability Real_Power
## 1:          NA        135
## 2:          NA        108
## 3:          30        126
## 4:          NA         95
## 5:          NA         70
## 6:          NA         81
```

11. Open typeChart_dt. Check classes. Fairy attacks are effective (i.e. Multiplier > 1) against which defense types?

```
sapply(typeChart_dt, class)
##        Attack       Defense Effectiveness    Multiplier
##   "character"   "character"   "character"     "numeric"
typeChart_dt[Attack == "FAIRY" & Multiplier > 1]
##    Attack  Defense   Effectiveness Multiplier
## 1:  FAIRY FIGHTING Super Effective          2
## 2:  FAIRY   DRAGON Super Effective          2
## 3:  FAIRY     DARK Super Effective          2
```

12. We use the sum of multiplier to evaluate the effectiveness of one type. Which is the most effective type, when attacking, against others? Which is the most effective, when defending, against others? Specifically, provide a sorted data.table with all the types and the sum of effectiveness for attack and another data table for defense.

```
attack_effectiveness <- typeChart_dt[,
                    .(Attack_Effectiveness = sum(Multiplier)), by = Attack]
setorder(attack_effectiveness, -Attack_Effectiveness)
attack_effectiveness
##       Attack Attack_Effectiveness
##  1:   GROUND                 21.0
##  2:     ROCK                 20.5
##  3:     FIRE                 20.0
##  4:      ICE                 20.0
##  5:    WATER                 19.5
##  6: FIGHTING                 19.5
##  7:   FLYING                 19.5
##  8:    FAIRY                 19.5
##  9:    STEEL                 19.0
## 10:    GHOST                 18.5
## 11:     DARK                 18.5
## 12:  PSYCHIC                 18.0
## 13: ELECTRIC                 17.5
## 14:    GRASS                 17.5
## 15:      BUG                 17.5
## 16:   DRAGON                 17.5
## 17:   POISON                 17.0
## 18:   NORMAL                 16.0

defense_effectiveness <- typeChart_dt[,
                    .(Defense_Effectiveness = sum(Multiplier)), by = Defense]
setorder(defense_effectiveness, -Defense_Effectiveness)
```

```
defense_effectiveness
##      Defense Defense_Effectiveness
##  1:      ICE                 21.5
##  2:    GRASS                 21.0
##  3:     ROCK                 21.0
##  4:  PSYCHIC                 20.0
##  5: FIGHTING                 19.5
##  6:      BUG                 19.5
##  7:   GROUND                 19.0
##  8:   DRAGON                 19.0
##  9:     DARK                 19.0
## 10:   FLYING                 18.5
## 11:   NORMAL                 18.0
## 12:     FIRE                 18.0
## 13:    WATER                 18.0
## 14: ELECTRIC                 17.5
## 15:   POISON                 17.5
## 16:    FAIRY                 17.5
## 17:    GHOST                 17.0
## 18:    STEEL                 15.0
```

Section IV. Joining all knowledge

1. Make a function called `best_attack()` that receives the name of 2 pokemons (attacker and defender) and returns the most powerful attack that the first pokemon should do (Power * Accuracy * Effectiveness Multiplier). For simplicity, consider that the defending pokemon only has its first type. Ignore 'Status' moves as they don't inflict any damage. Try some examples like: best_attack("Gastly", "Tangela"), best_attack("Pikachu", "Bellsprout")

```
best_attack = function(Poke1, Poke2){
  type1 = poke_dt[Name == Poke1, Type]
  type2 = poke_dt[Name == Poke2, Type][1]  # Consider only the 1st type

  attacks = moves_dt[Type %in% type1 & Category != "Status"]
  #attacks = moves_dt[Type %in% type1 & Category %in% c("Special", "Physical")]
  mult = typeChart_dt[Attack %in% type1 & Defense == type2, ]
  setnames(mult, "Attack", "Type")
  attacks = merge(attacks, mult[,.(Type, Multiplier)], by = "Type", sort = F)
  attacks[, power_mult := Real_Power * Multiplier]
  attacks[which_max(power_mult)]
}

best_attack("Gastly", "Tangela")
##      Type  Name Category Power Accuracy PP   TM
## 1: POISON Belch  Special   120       90 10 <NA>
##                             Effect Probability Real_Power Multiplier
## 1: Requires a held Berry to attack.          NA        108          2
##    power_mult
```

```
## 1:        216
best_attack("Pikachu", "Bellsprout")
##        Type        Name Category Power Accuracy PP   TM
## 1: ELECTRIC Volt Tackle Physical   120      100 15 <NA>
##                                                 Effect Probability
## 1: User receives recoil damage. May paralyze opponent.          10
##    Real_Power Multiplier power_mult
## 1:        120        0.5         60
```