Data Analysis and Visualization Exercise 2.2: Data import

Felix Brechtmann, Matthias Heinig, Vicente Yépez

30 October 2019

1 Flat file questions

1.1 Question

Read the titanic file (titanic.csv) and figure out who was the oldest surviving passenger of the titanic accident? Hint: `?subset`

```
tit_df <- read.csv("./extdata/titanic.csv")
head(tit_df)
##   pclass survived                                            name    sex
## 1      1        1                   Allen, Miss. Elisabeth Walton female
## 2      1        1                  Allison, Master. Hudson Trevor   male
## 3      1        0                    Allison, Miss. Helen Loraine female
## 4      1        0            Allison, Mr. Hudson Joshua Creighton   male
## 5      1        0 Allison, Mrs. Hudson J C (Bessie Waldo Daniels) female
## 6      1        1                             Anderson, Mr. Harry   male
##     age sibsp parch ticket     fare   cabin embarked boat body
## 1 29.00     0     0  24160 211.3375      B5        S    2   NA
## 2  0.92     1     2 113781 151.5500 C22 C26        S   11   NA
## 3  2.00     1     2 113781 151.5500 C22 C26        S        NA
## 4 30.00     1     2 113781 151.5500 C22 C26        S       135
## 5 25.00     1     2 113781 151.5500 C22 C26        S        NA
## 6 48.00     0     0  19952  26.5500     E12        S    3   NA
##                         home.dest
## 1                    St Louis, MO
## 2 Montreal, PQ / Chesterville, ON
## 3 Montreal, PQ / Chesterville, ON
## 4 Montreal, PQ / Chesterville, ON
## 5 Montreal, PQ / Chesterville, ON
## 6                    New York, NY
# restrict to survivors and get oldest; show name
subset( tit_df, survived==1 & age == max(age, na.rm=T), name)
##                                    name
## 15 Barkworth, Mr. Algernon Henry Wilson
```

1.2 Question

For the next two questions we simulate files as strings. They can be read, as if they where files.

A csv file has numbers as column names in the first row. Which parameter of read.table() needs to be adjusted to read the column names as they are in the csv?

```
tmp_tidy_table <- "1_colname,2_colname,3_colname
  3,4,5
  a,b,c"
read.csv(text = tmp_tidy_table)
##   X1_colname X2_colname X3_colname
## 1          3          4          5
## 2          a          b          c
```

```
# parameter `check.names`: a logical, tests for syntactically valid variable names
tidy_txt_df <- read.csv(text = tmp_tidy_table, check.names = FALSE)
tidy_txt_df
##   1_colname 2_colname 3_colname
## 1         3         4         5
## 2         a         b         c
```

1.3 Question

How to read the following table to have the `identical()` information as in `tidy_txt_df` from question above?

```
tmp_messy_table <- "# This line is just useless info

  1_colname,2_colname,3_colname
  3,4,5

  a,b,c"
```

```
# parameter `comment.char`: a character vector of length one
#   containing a single character or an empty string.
#   Use "" to turn off the interpretation of comments altogether.
# The parameter `blank.lines.skip` is TRUE by default.
messy_txt_df <- read.csv( text= tmp_messy_table, check.names = F, comment.char = '#')
identical(tidy_txt_df, messy_txt_df)
## [1] TRUE
```

2 Excel questions

2.1 Question

Read only `Name`, `Type` and `Total` columns for only the first 10 pokemons of the pokemon.xlsx file. Hint: take a look at the file using Excel or any other spreadsheet application.

```
library(readxl)
poke_file <- file.path('extdata/pokemon.xlsx')
poke_df <- read_excel(poke_file, sheet='Pokemon', range="B1:D11")
poke_df
## # A tibble: 10 x 3
##    Name          Type   Total
##    <chr>         <chr>  <dbl>
##  1 Bulbasaur     GRASS    318
##  2 Bulbasaur     POISON   318
##  3 Ivysaur       GRASS    405
##  4 Ivysaur       POISON   405
##  5 Venusaur      GRASS    525
##  6 Venusaur      POISON   525
##  7 Mega Venusaur GRASS    625
##  8 Mega Venusaur POISON   625
##  9 Charmander    FIRE     309
## 10 Charmeleon    FIRE     405
```

2.2 Question

Using the summer_olympic_medals.xlsx file, which athlete won most bronze medals?

```
oly_file <- file.path('extdata/summer_olympic_medals.xlsx')
```

```
library(readxl)
oly_df <- read_excel(oly_file, sheet='ALL MEDALISTS')
head(oly_df)
## # A tibble: 6 x 10
##   City  Edition Sport Discipline Athlete NOC   Gender Event Event_gender
##   <chr>   <dbl> <chr> <chr>      <chr>   <chr> <chr>  <chr> <chr>
## 1 Athe~    1896 Aqua~ Swimming   HAJOS,~ HUN   Men    100m~ M
## 2 Athe~    1896 Aqua~ Swimming   HERSCH~ AUT   Men    100m~ M
## 3 Athe~    1896 Aqua~ Swimming   DRIVAS~ GRE   Men    100m~ M
## 4 Athe~    1896 Aqua~ Swimming   MALOKI~ GRE   Men    100m~ M
## 5 Athe~    1896 Aqua~ Swimming   CHASAP~ GRE   Men    100m~ M
## 6 Athe~    1896 Aqua~ Swimming   CHOROP~ GRE   Men    1200~ M
## # ... with 1 more variable: Medal <chr>

# There are different solutions for this
oly_dt <- as.data.table(oly_df)
bronze <- oly_dt[Medal == "Bronze",]
```

```
# 1. Using table() function
bronze_counts <- table(subset(oly_dt, Medal == "Bronze", "Athlete"))
head(sort(bronze_counts, decreasing = T))
##
##                      NEMOV, Alexei               OTTEY-PAGE, Merlene
##                                  6                                 6
##                 SAVOLAINEN, Heikki            VAN ALMSICK, Franziska
##                                  6                                 6
##                BUSCHSCHULTE, Antje DE JONG, Adrianus Egbertus Willem
##                                  5                                 5
bronze_athl <- bronze_counts[ bronze_counts==max(bronze_counts)]
bronze_athl
##
##          NEMOV, Alexei    OTTEY-PAGE, Merlene     SAVOLAINEN, Heikki
##                      6                      6                      6
## VAN ALMSICK, Franziska
##                      6
# View their record
# subset(oly_df, Athlete %in% names(bronze_athl),
#    c('Athlete', 'Edition', "Sport", "Event", "Medal"))

# 2.Using aggregate() function
bronze.l <- as.data.table(aggregate(bronze,
                                    by=list(bronze$Athlete), FUN=length))
bronze.l[Medal == max(Medal),]
##                    Group.1 City Edition Sport Discipline Athlete NOC Gender
## 1:          NEMOV, Alexei    6       6     6          6       6   6      6
## 2:    OTTEY-PAGE, Merlene    6       6     6          6       6   6      6
## 3:     SAVOLAINEN, Heikki    6       6     6          6       6   6      6
## 4: VAN ALMSICK, Franziska    6       6     6          6       6   6      6
##    Event Event_gender Medal
## 1:     6            6     6
## 2:     6            6     6
## 3:     6            6     6
## 4:     6            6     6

# 3. Using dcast() function. More on this in Tidy Data lecture
ox <- dcast(oly_dt, Athlete ~ Medal)
oxx <- subset(ox, Bronze == max(Bronze))


# 4. Using .N command from data.table. More to this in Data Table lecture
bronze[, N := .N, by = Athlete]
bronze[N == max(N), unique(Athlete)]
## [1] "SAVOLAINEN, Heikki"     "OTTEY-PAGE, Merlene"
## [3] "VAN ALMSICK, Franziska" "NEMOV, Alexei"
```

2.3 Question

Are the columns `Gender` and `Event_gender` consistent? Find inconsistent gender entries.

```
# There was a male Bronze-medal winner in ladies marathon in 2000.
unique(oly_df$Gender)
## [1] "Men"   "Women"
unique(oly_df$Event_gender)
## [1] "M" "X" "W"
subset(oly_df, Gender=='Men' & Event_gender=='W')
## # A tibble: 1 x 10
##   City  Edition Sport Discipline Athlete NOC   Gender Event Event_gender
##   <chr>   <dbl> <chr> <chr>      <chr>   <chr> <chr>  <chr> <chr>
## 1 Sydn~    2000 Athl~ Athletics  CHEPCH~ KEN   Men    mara~ W
## # ... with 1 more variable: Medal <chr>
subset(oly_df, Gender=='Women' & Event_gender=='M')
## # A tibble: 0 x 10
## # ... with 10 variables: City <chr>, Edition <dbl>, Sport <chr>,
## #   Discipline <chr>, Athlete <chr>, NOC <chr>, Gender <chr>, Event <chr>,
## #   Event_gender <chr>, Medal <chr>
```

2.4 Question

Which country won most medals? Which country has the highest ratio of silver medals? Use the data in the country summary sheet starting at row 147 of the summer_olympic_medals.xlsx file.

```
# There is also a summary sheet for nations
nation_medal_df <- read_excel(oly_file, sheet='COUNTRY TOTALS', range="A147:F286")

head(nation_medal_df)
## # A tibble: 6 x 6
##   NOC         Country         Bronze  Gold Silver `Grand Total`
##   <chr>       <chr>            <dbl> <dbl>  <dbl>         <dbl>
## 1 Grand Total <NA>              9689  9850   9677         29216
## 2 USA         United States     1052  2088   1195          4335
## 3 URS         Soviet Union       584   838    627          2049
## 4 GBR         United Kingdom     505   498    591          1594
## 5 FRA         France             475   378    461          1314
## 6 GER         Germany            454   407    350          1211
# Remove Grand.Total row
nation_medal_df <- subset(nation_medal_df, !is.na(Country))
# Get max
subset(nation_medal_df, `Grand Total`==max(`Grand Total`))
## # A tibble: 1 x 6
##   NOC   Country       Bronze  Gold Silver `Grand Total`
##   <chr> <chr>          <dbl> <dbl>  <dbl>         <dbl>
## 1 USA   United States   1052  2088   1195          4335
```

```
nation_medal_df[,'silver_ratio'] <- with(nation_medal_df, Silver/`Grand Total`)
# alternatively:
# nation_medal_df[,'silver_ratio'] <- nation_medal_df$Silver/nation_medal_df$`Grand Total`
subset(nation_medal_df, silver_ratio==max(silver_ratio, na.rm=T))
## # A tibble: 13 x 7
##    NOC   Country               Bronze  Gold Silver `Grand Total` silver_ratio
##    <chr> <chr>                  <dbl> <dbl>  <dbl>         <dbl>        <dbl>
##  1 PAR   Paraguay                  NA    NA     17            17            1
##  2 SCG   Serbia                    NA    NA     14            14            1
##  3 NAM   Namibia                   NA    NA      4             4            1
##  4 SIN   Singapore                 NA    NA      4             4            1
##  5 SRI   Sri Lanka                 NA    NA      2             2            1
##  6 TAN   Tanzania                  NA    NA      2             2            1
##  7 VIE   Vietnam                   NA    NA      2             2            1
##  8 AHO   Netherlands Antilles*     NA    NA      1             1            1
##  9 CIV   Cote d'Ivoire             NA    NA      1             1            1
## 10 ISV   Virgin Islands*           NA    NA      1             1            1
## 11 SEN   Senegal                   NA    NA      1             1            1
## 12 SUD   Sudan                     NA    NA      1             1            1
## 13 TGA   Tonga                     NA    NA      1             1            1
```

2.5 Question

Which countries did participate, but without winning medals? Assume, that all countries listed in the IOC COUNTRY CODES sheet participated.

```
participants <- read_excel(oly_file, sheet='IOC COUNTRY CODES', range="A1:C202")
# workaround
# participants <- read.table(
#  sub('.xlsx','_IOC_COUNTRY_CODES.csv',oly_file),
#  sep=':', quote='@', header=T
#)
head(participants)
## # A tibble: 6 x 3
##   Country         `Int Olympic Committee code` `ISO code`
##   <chr>           <chr>                        <chr>
## 1 Afghanistan     AFG                          AF
## 2 Albania         ALB                          AL
## 3 Algeria         ALG                          DZ
## 4 American Samoa* ASA                          AS
## 5 Andorra         AND                          AD
## 6 Angola          ANG                          AO

## make sure to have proper variable names
colnames(participants) <- make.names(colnames(participants))

no_medals <- setdiff(participants$Int.Olympic.Committee.code, nation_medal_df$NOC)
length(no_medals)
## [1] 78
```

```
c(subset(participants, Int.Olympic.Committee.code %in% no_medals, "Country"))
## $Country
##  [1] "Albania"                          "American Samoa*"
##  [3] "Andorra"                          "Angola"
##  [5] "Antigua and Barbuda"              "Aruba*"
##  [7] "Bahrain"                          "Bangladesh"
##  [9] "Belize"                           "Benin"
## [11] "Bhutan"                           "Bolivia"
## [13] "Bosnia and Herzegovina"           "Botswana"
## [15] "British Virgin Islands"           "Brunei"
## [17] "Burkina Faso"                     "Cambodia"
## [19] "Cape Verde"                       "Cayman Islands*"
## [21] "Central African Republic"         "Chad"
## [23] "Comoros"                          "Congo"
## [25] "Congo, Dem Rep"                   "Cook Islands"
## [27] "Cyprus"                           "Dominica"
## [29] "East Timor (Timor-Leste)"         "El Salvador"
## [31] "Equatorial Guinea"                "Fiji"
## [33] "Gabon"                            "Gambia"
## [35] "Grenada"                          "Guam"
## [37] "Guatemala"                        "Guinea"
## [39] "Guinea-Bissau"                    "Honduras"
## [41] "Jordan"                           "Laos"
## [43] "Lesotho"                          "Liberia"
## [45] "Libya"                            "Liechtenstein"
## [47] "Madagascar"                       "Malawi"
## [49] "Maldives"                         "Mali"
## [51] "Malta"                            "Mauritania"
## [53] "Micronesia"                       "Monaco"
## [55] "Burma"                            "Nauru"
## [57] "Nepal"                            "Nicaragua"
## [59] "Oman"                             "Palau"
## [61] "Palestine, Occupied Territories"  "Papua New Guinea"
## [63] "Romania"                          "Rwanda"
## [65] "Saint Kitts and Nevis"            "Saint Lucia"
## [67] "Saint Vincent and the Grenadines" "Samoa"
## [69] "San Marino"                       "Sao Tome and Principe"
## [71] "Seychelles"                       "Sierra Leone"
## [73] "Solomon Islands"                  "Somalia"
## [75] "Swaziland"                        "Turkmenistan"
## [77] "Vanuatu"                          "Yemen"
```

3 SQL questions

3.1 Question

Connect to the `extdata/Northwind.sl3` SQLite data base (using the 'RSQLite' package). Inspect the data base tables using the 'dbListTables' and 'dbListFields' functions. Put together a SQL statement to retrieve a table that lists for all customers (name of the company, name of the contact person and city) all the products (name of the product) that they ordered. Execute the statement using 'dbGetQuery'. How many rows does this table have? Display the first 5 rows.

We provide the SQL statement here:

```
    "select customers.companyname, customers.contactname,
    customers.city, products.productname from customers inner join
    orders on customers.customerid = orders.customerid inner join
    `order details` on orders.orderid = `order details`.orderid inner
    join products on `order details`.productid = products.productid"
## [1] "select customers.companyname, customers.contactname,\n    customers.city, products.productname from c
```

```
library(RSQLite)
drv <- dbDriver("SQLite")
con <- dbConnect(drv, dbname="extdata/Northwind.sl3")
# Check all tables using
dbListTables(con)
##  [1] "Alphabetical list of products"  "Categories"
##  [3] "Current Product List"           "Customer and Suppliers by City"
##  [5] "CustomerCustomerDemo"           "CustomerDemographics"
##  [7] "Customers"                      "EmployeeTerritories"
##  [9] "Employees"                      "Order Details"
## [11] "Order Details Extended"         "Order Subtotals"
## [13] "Orders"                         "Orders Qry"
## [15] "Products"                       "Products Above Average Price"
## [17] "Products by Category"           "Region"
## [19] "Shippers"                       "Summary of Sales by Quarter"
## [21] "Summary of Sales by Year"       "Suppliers"
## [23] "Territories"                    "copy_of_customers"
# Check fields of a table
dbListFields(con, 'Customers')
##  [1] "CustomerID"   "CompanyName"  "ContactName"  "ContactTitle"
##  [5] "Address"      "City"         "Region"       "PostalCode"
##  [9] "Country"      "Phone"        "Fax"
dbListFields(con, 'Products')
##  [1] "ProductID"       "ProductName"     "SupplierID"      "CategoryID"
##  [5] "QuantityPerUnit" "UnitPrice"       "UnitsInStock"    "UnitsOnOrder"
##  [9] "ReorderLevel"    "Discontinued"

tab <- dbGetQuery(con, "select customers.companyname, customers.contactname,
                  customers.city, products.productname from customers inner join
                  orders on customers.customerid = orders.customerid inner join
                  `order details` on orders.orderid = `order details`.orderid inner
                  join products on `order details`.productid = products.productid")
nrow(tab)
## [1] 2155
tab[1:5,]
##                 CompanyName   ContactName       City
## 1 Vins et alcools Chevalier  Paul Henriot      Reims
## 2 Vins et alcools Chevalier  Paul Henriot      Reims
## 3 Vins et alcools Chevalier  Paul Henriot      Reims
## 4     Toms Spezialit\xe4ten Karin Josephs M\xfcnster
## 5     Toms Spezialit\xe4ten Karin Josephs M\xfcnster
##                     ProductName
## 1                Queso Cabrales
## 2 Singaporean Hokkien Fried Mee
## 3        Mozzarella di Giovanni
## 4                          Tofu
## 5         Manjimup Dried Apples
```

4 XML questions

4.1 Question

Load the XML document `plant_catalog.xml`. Use XPath and DOM functions to find out all unique element names in the document.

```
library(XML)
doc = xmlTreeParse("extdata/plant_catalog.xml", useInternal = TRUE)
#root = xmlRoot(doc)
unique(unlist(xpathApply(doc, "//*", xmlName)))
## [1] "CATALOG"      "PLANT"        "COMMON"       "BOTANICAL"
## [5] "ZONE"         "LIGHT"        "PRICE"        "AVAILABILITY"
```

Get all plants of zone 4 and transform the data into an R list. Hint: ‘xmlToList’

```
plant_list = xpathApply(doc, "//PLANT[ZONE = 4]", xmlToList)
length(plant_list)
## [1] 15
plant_list[[1]]
## $COMMON
## [1] "Bloodroot"
##
## $BOTANICAL
## [1] "Sanguinaria canadensis"
##
## $ZONE
## [1] "4"
##
## $LIGHT
## [1] "Mostly Shady"
```

```
##
## $PRICE
## [1] "$2.44"
##
## $AVAILABILITY
## [1] "031599"
```

4.2 Question

Read the HTML tables from the website https://www.skysports.com/premier-league-table into your workspace.

Which team is currently placed first in the premier league?

```
library(XML)
library(RCurl)
url <- "https://www.skysports.com/premier-league-table"
tables <- readHTMLTable(getURL(url))
table <- tables[[1]]
table[1,"Team"]
## [1] Liverpool
## 20 Levels: Arsenal Aston Villa Bournemouth ... Wolverhampton Wanderers
```

5 Prepare for next lecture

For data manipulation with the `data.table` package please read this intro.