

Assignment: ASSIGNMENT 2

Name: Praghuwanshi, Prashant

Date: 2021-06-20

Check your current working directory using `getwd()`

```
getwd()
```

List the contents of the working directory with the `dir()` function

```
dir()
```

If the current directory does not contain the data directory, set the working directory to project root folder (the folder should contain the data directory)

Use `setwd()` if needed

```
setwd("D:/MS_DataScience/DSC 520-Datastatistics.pdf/dsc520")
```

Load the file `data/tidynomicon/person.csv` to `person_df1` using `read.csv`

Examine the structure of `person_df1` using `str()`

```
person_df1 <- read.csv(file = "data/tidynomicon/person.csv", sep = ",", header=TRUE) str(person_df1)
```

R interpreted names as factors, which is not the behavior we want

Load the same file to `person_df2` using `read.csv` and setting `stringsAsFactors` to `FALSE`

Examine the structure of `person_df2` using `str()`

```
person_df2 <- read.csv(file = "data/tidynomicon/person.csv", sep = ",", header=TRUE, stringsAsFactors = FALSE) str(person_df2)
```

Read the file `data/scores.csv` to `scores_df`

Display summary statistics using the `summary()` function

```
scores_df <- summary(read.csv(file = "data/tidynomicon/person.csv", sep = ",", header=TRUE)) scores_df
```

Load the `readxl` library

```
library(readxl)
```

Using the `excel_sheets()` function from the `readxl` package,

list the worksheets from the file `data/G04ResultsDetail2004-11-02.xls`

```
excel_sheets("data/G04ResultsDetail2004-11-02.xls") ## Using the read_excel function, read the  
Voter Turnout sheet ## from the data/G04ResultsDetail2004-11-02.xls ## Assign the data to the  
voter_turnout_df1 ## The header is in the second row, so make sure to skip the first row ## Examine  
the structure of voter_turnout_df1 using str()
```

```
voter_turnout_df1 <- read_excel("data/G04ResultsDetail2004-11-02.xls", sheet= "Voter Turnout", skip=1)
str(voter_turnout_df1)
```

Using the read_excel() function, read the Voter Turnout sheet

from data/G04ResultsDetail2004-11-02.xls

Skip the first two rows and manually assign the columns using col_names

Use the names "ward_precint", "ballots_cast", "registered_voters", "voter_turnout"

Assign the data to the voter_turnout_df2

Examine the structure of voter_turnout_df2 using str()

```
voter_turnout_df2 <- read_excel("data/G04ResultsDetail2004-11-02.xls", sheet= "Voter Turnout",
skip=2, col_names = c("ward_precint", "ballots_cast", "registered_voters", "voter_turnout"))
str(voter_turnout_df2)
```

Load the DBI library

```
##install.packages("DBI") library(DBI) library(RSQLite) ## Create a database connection to
data/tidynomicon/example.db using the dbConnect() function ## The first argument is the database
driver which in this case is RSQLite::SQLite() ## The second argument is the path to the database
file ## Assign the connection to db variable db <- dbConnect(RSQLite::SQLite(), dbname =
"data/tidynomicon/example.db")
```

Query the Person table using the dbGetQuery function and the

SELECT * FROM PERSON; SQL statement

Assign the result to the person_df variable

Use head() to look at the first few rows of the person_df dataframe

```
person_df <- dbGetQuery(db, "SELECT * FROM PERSON") head(person_df, 6)
```

List the tables using the dbListTables() function

Assign the result to the table_names variable

```
table_names <- dbListTables(db)
```

Read all of the tables at once using the lapply function and assign the result to the tables variable

Use table_names, dbReadTable, and conn = db as arguments

Print out the tables

```
table_names_fun <- function(tablename) { fetchdata <- dbReadTable(db, tablename) return(data.frame(fetchdata))
} tables <- lapply(table_names, fun) tables
```

Use the dbDisconnect function to disconnect from the database

```
dbDisconnect(db)
```

Import the jsonlite library

```
library(jsonlite)
```

Convert the scores_df dataframe to JSON using the toJSON() function

```
toJSON(data.frame(rbind(scores_df)))
```

Convert the scores dataframe to JSON using the toJSON() function with the pretty=TRUE option

```
toJSON(data.frame(rbind(scores_df)), pretty = TRUE)
```