DSC520_Week2Assignment_02

Henry Pham

2023-03-26

```r
## Check your current working directory using `getwd()`

getwd()
```

```
## [1] "C:/R/DSC520"
```

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

dir("C:/R/DSC520")
```

```
##  [1] "assignment_00_LastnameFirstname.R.txt"
##  [2] "assignment_01_LastnameFirstname.R.txt"
##  [3] "assignment_02_LastnameFirstname.R.txt"
##  [4] "data"
##  [5] "DESCRIPTION"
##  [6] "DSC520.Rproj"
##  [7] "DSC520_Pham_Week2Assignment_00.pdf"
##  [8] "DSC520_Pham_Week2Assignment_01.pdf"
##  [9] "DSC520_Week2Assignment.R"
## [10] "DSC520_Week2Assignment_00.docx"
## [11] "DSC520_Week2Assignment_00.Rmd"
## [12] "DSC520_Week2Assignment_01.docx"
## [13] "DSC520_Week2Assignment_01.R"
## [14] "DSC520_Week2Assignment_01.Rmd"
## [15] "DSC520_Week2Assignment_02.docx"
## [16] "DSC520_Week2Assignment_02.R"
## [17] "DSC520_Week2Assignment_02.Rmd"
## [18] "example.db"
## [19] "G04ResultsDetail2004-11-02.xls"
## [20] "man"
## [21] "NAMESPACE"
## [22] "person.csv"
## [23] "R"
## [24] "scores.csv"
```

```r
## 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("/home/jdoe/Workspaces/dsc520")

# Check if the data directory exists in the current working directory
if (!file.exists("data"))
```

```
{
# set data working directory
  setwd("C:/R/DSC520/data")
}

## Load the file `data/tidynomicon/person.csv` to `person_df1` using
`read.csv`
## Examine the structure of `person_df1` using `str()`

# Load the person.csv file to person_df1
person_df1 <- read.csv("data/tidynomicon/person.csv")

# Examine the structure of person_df1
str(person_df1)

## 'data.frame':    5 obs. of  1 variable:
##  $ person_id.personal_name.family_name: chr  "dyer,William,Dyer"
"pb,Frank,Pabodie" "lake,Anderson,Lake" "roe,Valentina,Roerich" ...

## 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()`

# Load the person.csv file to person_df2, setting stringsAsFactors to FALSE
person_df2 <- read.csv("data/tidynomicon/person.csv", stringsAsFactors =
FALSE)

# Examine the structure of person_df2
str(person_df2)

## 'data.frame':    5 obs. of  1 variable:
##  $ person_id.personal_name.family_name: chr  "dyer,William,Dyer"
"pb,Frank,Pabodie" "lake,Anderson,Lake" "roe,Valentina,Roerich" ...

## Read the file `data/scores.csv` to `scores_df`
## Display summary statistics using the `summary()` function

# Read the scores.csv file to scores_df
scores_df <- read.csv("data/scores.csv")

# Display summary statistics using the summary() function
summary(scores_df)

##  Count.Score.Section
##  Length:38
##  Class :character
##  Mode  :character
## 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`

# Load the readxl library
library(readxl)

# List the worksheets in the G04ResultsDetail2004-11-02.xls file
sheets <- excel_sheets("data/G04ResultsDetail2004-11-02.xls")
sheets

##  [1] "Instructions"          "Voter Turnout"         "President"
##  [4] "House of Rep"          "Co Clerk"              "Co Reg Deeds"
##  [7] "Co Public Defender"    "Co Comm 1"             "Co Comm 3"
## [10] "Co Comm 5"             "Co Comm 7"             "St Bd of Ed 2"
## [13] "St Bd of Ed 4"         "Legislature 5"         "Legislature 7"
## [16] "Legislature 9"         "Legislature 11"        "Legislature 13"
## [19] "Legislature 23"        "Legislature 31"        "Legislature 39"
## [22] "MCC 1"                 "MCC 2"                 "MCC 3"
## [25] "MCC 4"                 "OPPD"                  "MUD"
## [28] "NRD 3"                 "NRD 5"                 "NRD 7"
## [31] "NRD 9"                 "OPS 2"                 "OPS 4"
## [34] "OPS 6"                 "OPS 8"                 "OPS 10"
## [37] "OPS 11"                "OPS 12"                "ESU 2"
## [40] "ESU 3"                 "Arlington Sch 24"      "Bennington Sch 59"
## [43] "Elkhorn Sch 10"        "Fremont Sch 1"         "Ft Calhoun Sch 3"
## [46] "Gretna Sch 37"         "Millard Sch 17"        "Ralston Sch 54"
## [49] "Valley Sch 33"         "Waterloo Sch 11"       "Bennington Mayor"
## [52] "Elkhorn Mayor"         "Valley Mayor"          "Ralston Mayor"
## [55] "Ralston Library Bd"    "Bennington City Cnc 1" "Bennington City Cnc
2"
## [58] "Elkhorn City Cnc A"    "Elkhorn City Cnc B"    "Elkhorn City Cnc C"
## [61] "Ralston City Cnc 1"    "Ralston City Cnc 2"    "Ralston City Cnc 6"
## [64] "Waterloo Bd Trustees"  "Valley City Cnc"       "Amendment 1"
## [67] "Amendment 2"           "Amendment 3"           "Amendment 4"
## [70] "Initiative 417"        "Initiative 418"        "Initiative 419"
## [73] "Initiative 420"

## 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()`

# Load the readxl library
library(readxl)
```

```
# Read the Voter Turnout sheet from the G04ResultsDetail2004-11-02.xls file
voter_turnout_df1 <- read_excel("data/G04ResultsDetail2004-11-02.xls", sheet
= "Voter Turnout", skip = 1)

# Examine the structure of voter_turnout_df1
str(voter_turnout_df1)

## tibble [342 × 4] (S3: tbl_df/tbl/data.frame)
##  $ Ward Precinct    : chr [1:342] "01-01" "01-02" "01-03" "01-04" ...
##  $ Ballots Cast     : num [1:342] 421 443 705 827 527 323 358 410 440 500
...
##  $ Registered Voters: num [1:342] 678 691 1148 1308 978 ...
##  $ Voter Turnout    : num [1:342] 0.621 0.641 0.614 0.632 0.539 ...

## 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()`

# Load the readxl library
library(readxl)

# Manually assign column names
col_names <- c("ward_precint", "ballots_cast", "registered_voters",
"voter_turnout")

# Read the Voter Turnout sheet from the G04ResultsDetail2004-11-02.xls file
and manually assign column names
voter_turnout_df2 <- read_excel("data/G04ResultsDetail2004-11-02.xls", sheet
= "Voter Turnout", skip = 2, col_names = col_names)

# Examine the structure of voter_turnout_df2
str(voter_turnout_df2)

## tibble [342 × 4] (S3: tbl_df/tbl/data.frame)
##  $ ward_precint     : chr [1:342] "01-01" "01-02" "01-03" "01-04" ...
##  $ ballots_cast     : num [1:342] 421 443 705 827 527 323 358 410 440 500
...
##  $ registered_voters: num [1:342] 678 691 1148 1308 978 ...
##  $ voter_turnout    : num [1:342] 0.621 0.641 0.614 0.632 0.539 ...

## Load the `DBI` library

library(DBI)

## 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

# Load the DBI library
library(DBI)

# Create a database connection to example.db using RSQLite::SQLite()
db <- dbConnect(RSQLite::SQLite(), "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

# Load the DBI library
library(DBI)

# Create a database connection to example.db using RSQLite::SQLite()
db <- dbConnect(RSQLite::SQLite(), "data/tidynomicon/example.db")

# Query the Person table using the dbGetQuery() function
person_df <- dbGetQuery(db, "SELECT * FROM PERSON;")

# Look at the first few rows of the person_df dataframe
head(person_df)

##   person_id personal_name family_name
## 1      dyer       William        Dyer
## 2        pb         Frank     Pabodie
## 3      lake      Anderson        Lake
## 4       roe     Valentina     Roerich
## 5  danforth         Frank    Danforth

## List the tables using the `dbListTables()` function
## Assign the result to the `table_names` variable

# Load the DBI library
library(DBI)

# Create a database connection to example.db using RSQLite::SQLite()
db <- dbConnect(RSQLite::SQLite(), "data/tidynomicon/example.db")

# List the tables in the database using the dbListTables() function
table_names <- dbListTables(db)
```

```
# Print the table names
table_names

## [1] "Measurements" "Person"       "Site"         "Visited"

## 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

# Load the DBI library
library(DBI)

# Create a database connection to example.db using RSQLite::SQLite()
db <- dbConnect(RSQLite::SQLite(), "data/tidynomicon/example.db")

# List the tables in the database using the dbListTables() function
table_names <- dbListTables(db)

# Use lapply to read all tables at once
tables <- lapply(table_names, dbReadTable, conn = db)

## Warning: Column `reading`: mixed type, first seen values of type real,
coercing
## other values of type string

# Print out the tables
tables

## [[1]]
##    visit_id person_id quantity reading
## 1       619      dyer      rad    9.82
## 2       619      dyer      sal    0.13
## 3       622      dyer      rad    7.80
## 4       622      dyer      sal    0.09
## 5       734        pb      rad    8.41
## 6       734      lake      sal    0.05
## 7       734        pb     temp  -21.50
## 8       735        pb      rad    7.22
## 9       735      <NA>      sal    0.06
## 10      735      <NA>     temp  -26.00
## 11      751        pb      rad    4.35
## 12      751        pb     temp  -18.50
## 13      751      lake      sal    0.00
## 14      752      lake      rad    2.19
## 15      752      lake      sal    0.09
## 16      752      lake     temp  -16.00
## 17      752       roe      sal   41.60
## 18      837      lake      rad    1.46
## 19      837      lake      sal    0.21
## 20      837       roe      sal   22.50
## 21      844       roe      rad   11.25
## 
## [[2]]
##   person_id personal_name family_name
## 1      dyer       William        Dyer
## 2        pb         Frank     Pabodie
## 3      lake      Anderson        Lake
## 4       roe     Valentina     Roerich
## 5  danforth         Frank    Danforth
## 
## [[3]]
##   site_id latitude longitude
## 1    DR-1   -49.85   -128.57
## 2    DR-3   -47.15   -126.72
## 3   MSK-4   -48.87   -123.40
## 
## [[4]]
##   visit_id site_id visit_date
## 1      619    DR-1 1927-02-08
## 2      622    DR-1 1927-02-10
## 3      734    DR-3 1930-01-07
## 4      735    DR-3 1930-01-12
## 5      751    DR-3 1930-02-26
## 6      752    DR-3       <NA>
## 7      837   MSK-4 1932-01-14
## 8      844    DR-1 1932-03-22
```

```r
## Use the `dbDisconnect` function to disconnect from the database

# Load the DBI library
library(DBI)

# Create a database connection to example.db using RSQLite::SQLite()
db <- dbConnect(RSQLite::SQLite(), "data/tidynomicon/example.db")

# Disconnect from the database
dbDisconnect(db)

## Import the `jsonlite` library

library(jsonlite)

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

# Import the jsonlite library
library(jsonlite)

# Convert the scores_df dataframe to JSON
scores_json <- toJSON(scores_df)
```

```
# Print the JSON output
cat(scores_json)

##
[{"Count.Score.Section":"10,200,Sports"},{"Count.Score.Section":"10,205,Sport
s"},{"Count.Score.Section":"20,235,Sports"},{"Count.Score.Section":"10,240,Sp
orts"},{"Count.Score.Section":"10,250,Sports"},{"Count.Score.Section":"10,265
,Regular"},{"Count.Score.Section":"10,275,Regular"},{"Count.Score.Section":"3
0,285,Sports"},{"Count.Score.Section":"10,295,Regular"},{"Count.Score.Section
":"10,300,Regular"},{"Count.Score.Section":"20,300,Sports"},{"Count.Score.Sec
tion":"10,305,Sports"},{"Count.Score.Section":"10,305,Regular"},{"Count.Score
.Section":"10,310,Regular"},{"Count.Score.Section":"10,310,Sports"},{"Count.S
core.Section":"20,320,Regular"},{"Count.Score.Section":"10,305,Regular"},{"Co
unt.Score.Section":"10,315,Sports"},{"Count.Score.Section":"20,320,Regular"},
{"Count.Score.Section":"10,325,Regular"},{"Count.Score.Section":"10,325,Sport
s"},{"Count.Score.Section":"20,330,Regular"},{"Count.Score.Section":"10,330,S
ports"},{"Count.Score.Section":"30,335,Sports"},{"Count.Score.Section":"10,33
5,Regular"},{"Count.Score.Section":"20,340,Regular"},{"Count.Score.Section":"
10,340,Sports"},{"Count.Score.Section":"30,350,Regular"},{"Count.Score.Sectio
n":"20,360,Regular"},{"Count.Score.Section":"10,360,Sports"},{"Count.Score.Se
ction":"20,365,Regular"},{"Count.Score.Section":"20,365,Sports"},{"Count.Scor
e.Section":"10,370,Sports"},{"Count.Score.Section":"10,370,Regular"},{"Count.
Score.Section":"20,375,Regular"},{"Count.Score.Section":"10,375,Sports"},{"Co
unt.Score.Section":"20,380,Regular"},{"Count.Score.Section":"10,395,Sports"}]

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

# Import the jsonlite library
library(jsonlite)

# Convert the scores_df dataframe to pretty JSON
scores_json_pretty <- toJSON(scores_df, pretty = TRUE)

# Print the pretty JSON output
cat(scores_json_pretty)

## [
##   {
##     "Count.Score.Section": "10,200,Sports"
##   },
##   {
##     "Count.Score.Section": "10,205,Sports"
##   },
##   {
##     "Count.Score.Section": "20,235,Sports"
##   },
##   {
##     "Count.Score.Section": "10,240,Sports"
##   },
##   {
##     "Count.Score.Section": "10,250,Sports"
##   },
##   {
##     "Count.Score.Section": "10,265,Regular"
##   },
##   {
##     "Count.Score.Section": "10,275,Regular"
##   },
##   {
##     "Count.Score.Section": "30,285,Sports"
##   },
##   {
##     "Count.Score.Section": "10,295,Regular"
##   },
##   {
##     "Count.Score.Section": "10,300,Regular"
##   },
##   {
##     "Count.Score.Section": "20,300,Sports"
##   },
##   {
##     "Count.Score.Section": "10,305,Sports"
##   },
##   {
##     "Count.Score.Section": "10,305,Regular"
##   },
##   {
##     "Count.Score.Section": "10,310,Regular"
##   },
##   {
##     "Count.Score.Section": "10,310,Sports"
##   },
##   {
##     "Count.Score.Section": "20,320,Regular"
##   },
##   {
##     "Count.Score.Section": "10,305,Regular"
##   },
##   {
##     "Count.Score.Section": "10,315,Sports"
##   },
##   {
##     "Count.Score.Section": "20,320,Regular"
##   },
##   {
##     "Count.Score.Section": "10,325,Regular"
##   },
##   {
```

```
##     "Count.Score.Section": "10,325,Sports"
##   },
##   {
##     "Count.Score.Section": "20,330,Regular"
##   },
##   {
##     "Count.Score.Section": "10,330,Sports"
##   },
##   {
##     "Count.Score.Section": "30,335,Sports"
##   },
##   {
##     "Count.Score.Section": "10,335,Regular"
##   },
##   {
##     "Count.Score.Section": "20,340,Regular"
##   },
##   {
##     "Count.Score.Section": "10,340,Sports"
##   },
##   {
##     "Count.Score.Section": "30,350,Regular"
##   },
##   {
##     "Count.Score.Section": "20,360,Regular"
##   },
##   {
##     "Count.Score.Section": "10,360,Sports"
##   },
##   {
##     "Count.Score.Section": "20,365,Regular"
##   },
##   {
##     "Count.Score.Section": "20,365,Sports"
##   },
##   {
##     "Count.Score.Section": "10,370,Sports"
##   },
##   {
##     "Count.Score.Section": "10,370,Regular"
##   },
##   {
##     "Count.Score.Section": "20,375,Regular"
##   },
##   {
##     "Count.Score.Section": "10,375,Sports"
##   },
##   {
##     "Count.Score.Section": "20,380,Regular"
##   },
##   {
##     "Count.Score.Section": "10,395,Sports"
##   }
## ]
```