**5.1 Lectures**

Importing Data in R: Excel, Json and Relational Database systems are various types of format in which data can be present.

Two ways data is stored: 1. Flat file where it is discrete.  
 2. Relational database or non-discrete format.

Inbuilt datasets in R can simply be called by their names.

Eg. To summarize the ‘BOD’ dataset we use the following syntax

* summary(BOD)

Connecting R to external data source- discrete files

* In the upper right quadrant in R studio, we use the ‘Import Dataset’ option and specify the location of the data from where to read. We adjust the parameters according to the data and import it into R studio. We can store it into a variable and run the functions we have run on already existing dataframes.
* For importing spreadsheets, we have to use a few R packages like gdata, RODBC, xlxReadWrite. We have to install the package and then use the library command to use the packages.  
  install.packages(“PackageName”)  
  library(“PackageName”)
* To import a spreadsheet from a URL we use the following R commands.  
  library(gdata)  
  testFrame <- read.xls(“URL\_to\_read”)  
  We can then run the same operations as any other dataframe on testFrame.  
  When we run View(testFrame) with additional columns having NAs.  
  The data set needs to be cleansed and transformed before inferring from it.   
  When we require a particular cleanse or transformation on different types of columns, it makes sense to create a single function that does the transformation and call it multiple times.

**5.2 SQL from R**

SQL databases are non-discrete database access. Where SQL has all the data on the server and we use R to extract some or all of the data. We use different packages for database connectivity.

We create a connection to the database, then get information about the tables we would like to extract and then extract the data from those tables. The output is a data frame.

* sapply(variableName, function)  
  This performs the function given as a parameter on the variableName.
* tapply(SummaryVariable, GroupVariable, function)  
  This performs the function on the summaryVariable grouped by GroupVariable.

**5.3 Json**

Json: Java Script Object Notation.

Json is a web based standard to access information on servers. It is not stored in the relational database format. R is not designed to handle database management and should be restricted to analytics.

To use Google API and storing the output, we write a function:

* MakeGeoURL <- function(address)  
  {  
  root <- ‘GoogleAPI’  
  url <- paste(root, “json?address=”,address, “&sensor=false”, sep= “”)  
  return(URLencode(url))  
  }

To isolate the lat and long coordinates from the API, we write another function:

* Addr2LatLong <- function(address)  
  {  
  url <-MakeGeoURL(address)  
    
  apiResult <-getURL(url)  
    
  geoStruct <-fromJSON(apiResult, simplify = FALSE)  
  lat <- NA  
  lng <- NA  
  try(lat <- geoStruct$results[[1]$geometry$location$lat)  
  try(lng <- geoStruct$results[[1]$geometry$location$lng)  
  return(c(lat,lng))  
  }

**5.4 Data Science Roundtable 3**

Many different analyses can be made by data. We can understand a lot by putting the data into context. In the airline example, the data can be grouped into different categories of likelihood to cancel or fly.

**Questions from the videos:**

* Why are reading spreadsheets sometimes not practical/appropriate?
* Sometimes the data is not in a tabular form or a flat file and it cannot be represented in a flat file either. In such cases, reading spreadsheets will not work.
* Why is the data available via JSON?
* For a bulk of data it is easier to store it in a central location and access it by API, which gives the data available in JSON format.
* What are the other alternatives to JSON?
* Spreadsheets can be used instead of JSON, but that process will become cumbersome.
* Why did citibike make the data available?
* It was open sourced so that they could let other people analyse the data and let other people find business angles for themselves.

**My Question :**

* Can we save the data after transformation back as a csv file?