# **Class 607: Data Load and Processing**

1. **Data Sources**
2. **SQL Databases** such as Access, SQL Server, and MySQL
3. **Web** – html, xlm, json, and API’s
4. **Csv** and other text files

Note: Data is generally classified into two types

1. **Structured**, easy to read well organized, such as data in SQL tables
2. **Unstructured**, hard to read, such as data found on Web pages
3. **SQL** - language to manipulate the normalized data
4. Typical SQL DB has tight **security**; different **privileges**; **passwords**
5. **Tables** – collection of data; should be normalized
6. **Primary key** – index of the unique table rows
7. **Secondary key** – primary key in linked table
8. Type of **relationships** – one-to-one; many-to-many; one-to-many
9. **Create** – SQL command to create db, table, view, and so on
10. **Select** – SQL command to select need columns from the table
11. **Join** – to join multiple tables; **inner join, outer join; left/right joins**
12. **Where** – to subset a table
13. **Union** – to append tables/views
14. **Update** – to modify a row
15. **Insert** – to insert a row/rows
16. **Case when … else … end as** – type of if statement
17. Other commands
18. **R** – language to work with data
19. Needs multiple **Librarie**s to work:
20. **load.packages**(“plyr”); “dplyr”, “XML”, “stringr”, and million others
21. **library**(plyr)
22. Case sensitive!
23. Uses **NA** for unknown
24. **::** - indicates what library is used (optional); dplyr::arrange
25. Reads **files** in:
26. **CSV** and other text files (read.csv)
27. Can connect to **SQL DB’s** (such as MySQL)
28. Can read **html, json**, and **xml** files
29. Stores **data** in:
30. **Vectors** (myvec <- c(1:10)) – the same format
31. **Lists** (myList <- list[1,”a”,NA]) – could be different formats
32. **Dataframes** (myDF <- data.frame(???))
33. Matrix
34. **Typeof**(myObject) – will return the type of an object
35. **Is.numeric**(myVec) - returns Boolean TRUE/FALSE
36. **as.numeric**(myVec) – converts to numeric
37. **Factors** – is present list of strings. Needs extra work to modify
38. **Dataframes**:
39. **myDF$myCol**; will return column “myCol” of “myDF” dataframe
40. **myDF[1,]** – will return a first row
41. **myDF[,1]** – will return a first column
42. **summary(myDF)** – will return summary of dataframe; all fields, max value, min value, min, standard variation, distribution, etc
43. **head**(myDF) – will return first 6 rows
44. **dim**(myDF) – will return # of rows and of columns; 3 (rows) 5 (columns)
45. **colnames**(myDF) <- c(“Col1”,”Col2”,…”ColN”); will name/rename the columns
46. newDF<-**subset**(myDF, “Col1”==NA); subsets DF based on definition
47. **table**(myDF$myCol) – will return frequencies of values of column “myCol”
48. **Strings**
49. Library – **stringr**
50. All commands start with **str\_**
51. We need to find pattern using complicated syntax
52. **str\_substring** (“Mikhail”,2,5). Returns “ikha”
53. **[abc]** – looks for “abc”
54. **Functions** – very confusing
55. **Logical operations; ifelse** – very confusing syntax
56. **==** checks for equality
57. **Plots** – very important !!!!
58. **“Ggplot”** library
59. **Plot**(x,y) – regular plot
60. **Hist**(x) - histogram !
61. **Box**(x, y) – boxplot
62. **Bar(myDF$myCol1, myDF$myCol2) –** bar chart
63. **Rmarkdown** – a tool in R to generate presentable code and output
64. Files for output are **HTML, PDF**, and others
65. Often stored in **Rpubs**
66. It has multiple **thyme**
67. **Toc** is used to generate Table of contents
68. **Web**
69. Pages are created in **HTML**, and enhanced by **Java**
70. **HTML** has **tables**, structured as any regular table
71. **HTTP** is used to communicated between a server and web page
72. **HTML** and **XML** use tags; they are markdown languages
73. **CSS** – language to code style of webpage
74. **Xpath** is used to work with XML
75. **URL** - a web page address
76. **JSON** – data format based on Java, growing in popularity; used by many API for data transfer
77. **Web scraping** in R – used to collect useful info from the Web
78. **Download.file()** – R function
79. **getHTMLTable()** (library XML) – R function
80. **Libraries**
81. Stringr
82. XML
83. RCurl
84. **Xpath** -?
85. **API** interface in R
86. Using the following R **commands**:
87. getURL
88. Basic commands used to read HTML, PDF, and FTP files from Web
89. Selenium to read Java enhanced web pages
90. **Rvest** – new library ???; inspired by beautiful soap (Python)
91. **SelectorGadget –** CSS tool, CSS used to create style of webpage!
92. **Visualization** (!!!!)
93. **Histograms** – frequency of a variable; R function - hist(myDF$myvar)
94. **Scatter Plot –** distribution of two variables; R function - scatter(myDF$mycol1, myDF$mycol2); good for many to many
95. **Bar plot** – good is for “one to one” and if x is a factor (?) and y is a numeric variable; R function bar(myDF$mycol1, myDF$mycol2)
96. **Ggplot2** – great library to do visualizations
97. **General Remarks**
98. **Data validation** is very important; we need to get primary sources whatever possible; look for reasonability
99. **Proxy** – something that might approximate the result I need; body temperature as a proxy for inflammation/virus/bacteria status of the body; EKG is a proxy for heart status; GDP per capita is a proxy for country livability status; is based on correlation
100. **Data science process condensed (taken from R for Data Science)**
101. Load data, to get data in R
102. Tidy up data, to make it a basic table, columnsXrows
103. Transform data, to create new variables, to drop others
104. To create model
105. Visualize
106. Present
107. **GitHub** – tool to share programs with version control
108. Need one local copy and another web based; they need to be synced
109. Has **repository**, where projects are stored
110. Repository could be **fork**ed to another account
111. **Tidy**ing up data
112. **Wide** – many columns
113. **Long** – fewer columns
114. **Tydyr** – R package
115. **Dplyr** – package to transform data
116. **Plyr** – older version
117. **Table\_dr** – creates easy to read table
118. **%>%** - pipe, allows to write multiple statements in one line, DF or Table is not repeat; myTbl %>% arrange(myCol1)
119. **Arrange –** sorts data
120. **Summary** – sums up dataset/table
121. **Count** – counts dataset/table
122. **Web Scrape** in R
123. **Web Scrape** Steps (taken from book “Automated Data Collection in R”, Ch 9)
124. **ID Website** (NYT API)
125. **Choose/develop strategy** (R interface)
126. **Retrieve the data** (JSON files)