Unit 1 – Data Import

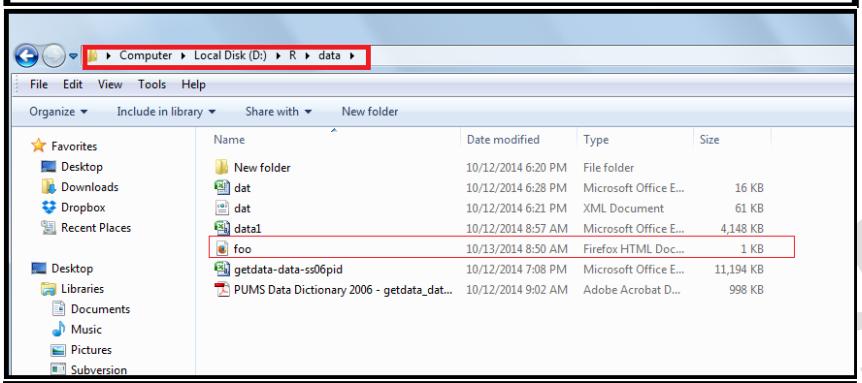
Contents

1. Create directory, if it does not exist

```
> #  ==================================================================
> #  Check the existence of a directory and create if it does not exist
> #  ==================================================================
> main_dir<-"D:/R"
> sub_dir<-"data"
> #  ==================================================================
> if(file.exists(sub_dir)) {
+ setwd(file.path(main_dir, sub_dir))
+ }else {
+ #
+ dir.create(file.path(main_dir, sub_dir))
+ setwd(file.path(main_dir, sub_dir))
+ }
> #  ==================================================================
> getwd()
[1] "D:/R/data"
```

2. Download a file

```
> download.file("http://www.r-project.org/index.html", destfile="foo.html")
trying URL 'http://www.r-project.org/index.html'
Content type 'text/html' length 788 bytes
opened URL
downloaded 788 bytes
> file.show("foo.html")
> getwd()
[1] "D:/R/data"
```



RGui (32-bit) - [R Information]

File Edit Windows

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">
<html>
<head>
<title>The R Project for Statistical Computing</title>
<link rel="icon" href="favicon.ico" type="image/x-icon">
<link rel="shortcut icon" href="favicon.ico" type="image/x-icon">
<link rel="stylesheet" type="text/css" href="R.css">
</head>

<FRAMESET cols="1*, 4*">
<FRAMESET rows="120, 1*">
<FRAME src="logo.html" name="logo" frameborder=0>
<FRAME src="navbar.html" name="contents" frameborder=0>
</FRAMESET>
<FRAME src="main.shtml" name="banner" frameborder=0>
<noframes>
<h1>The R Project for Statistical Computing</h1>

Your browser seems not to support frames,
here is the <A href="navbar.html">contents page</A> of the R Project's
website.
</noframes>
</FRAMESET>
```

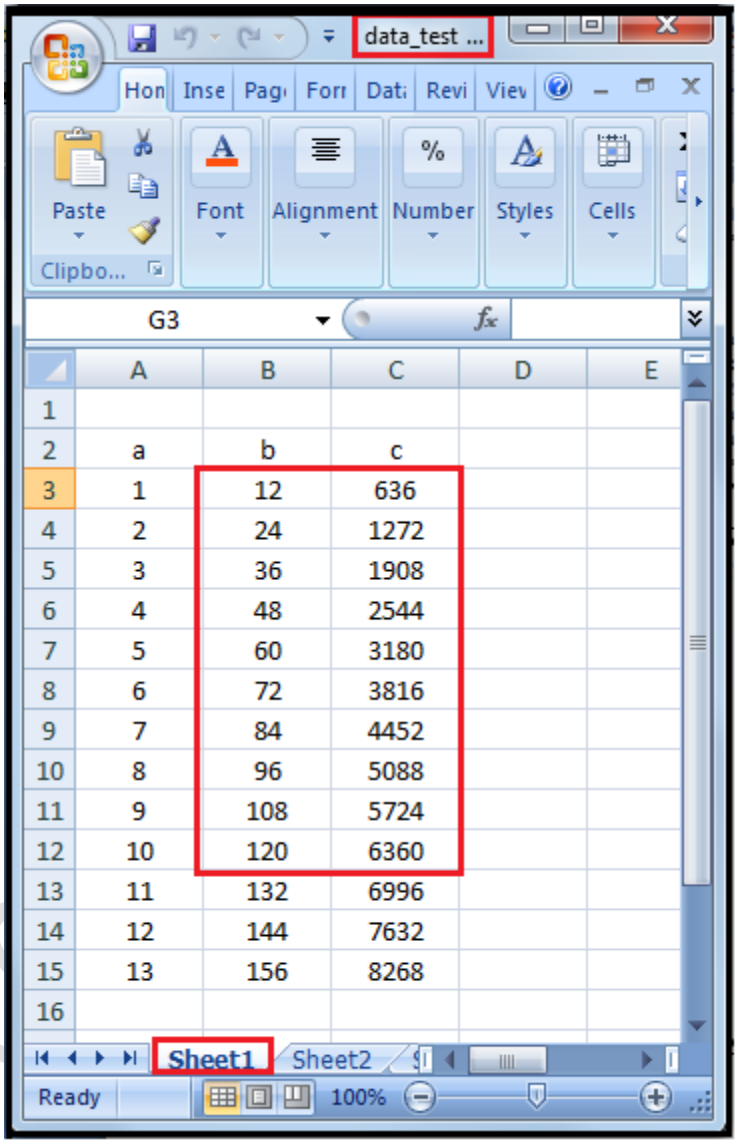
3. Read an excel file

RGui (32-bit) - [Untitled - R Editor]

File Edit Packages Windows Help

```
#  ==========================================================================================
# to read an excel file in xlsx format
#  ==========================================================================================
#
library(gdata)
#
dest_file   = "D:/R/data/data_test.xlsx"
dat_res     <-    read.xls(dest_file,sheet="Sheet1",nrows=10,blank.lines.skip=T)
dat         = dat_res[,2:3]
sumx        <-    sum(dat$b*dat$c,na.rm=T)
cat("\n a --> ",sumx,"\n")
#  ==========================================================================================
```

```
  a -->  2938320
> dat
     b    c
1   12  636
2   24 1272
3   36 1908
4   48 2544
5   60 3180
6   72 3816
7   84 4452
8   96 5088
9  108 5724
10 120 6360
```



	A	B	C
1			
2	a	b	c
3	1	12	636
4	2	24	1272
5	3	36	1908
6	4	48	2544
7	5	60	3180
8	6	72	3816
9	7	84	4452
10	8	96	5088
11	9	108	5724
12	10	120	6360
13	11	132	6996
14	12	144	7632
15	13	156	8268
16			

4. JSON files

JSON

JSON or JavaScript Object Notation, is an open standard format that uses human-readable text to transmit data objects consisting of attribute–value pairs. It is used primarily to transmit data between a server and web application, as an alternative to XML.

Although originally derived from the JavaScript scripting language, JSON is a language-independent data format. Code for parsing and generating JSON data is readily available in a large variety of programming languages.

The JSON format was originally specified by Douglas Crockford. It is currently described by two competing standards, RFC 7159 and ECMA-404. The ECMA standard is minimal, describing only the allowed grammar syntax, whereas the RFC also provides some semantic and security considerations.[2] The official Internet media type for JSON is application/json. The JSON filename extension is .json.

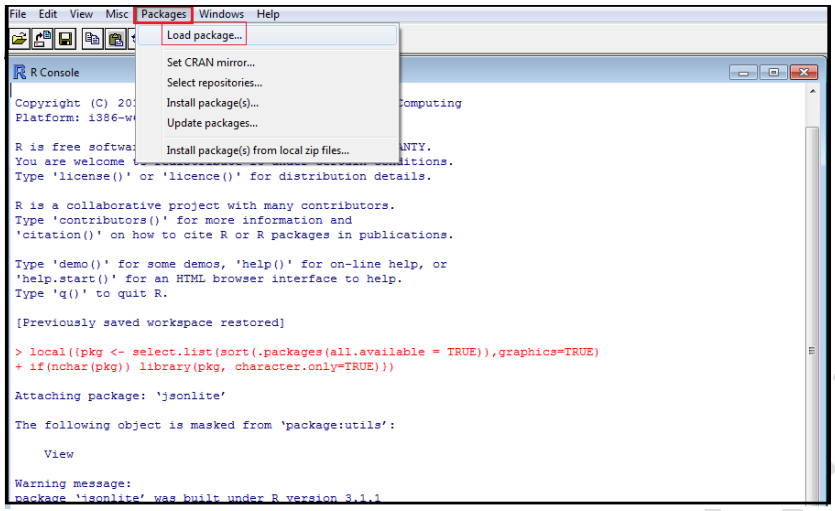
Features:

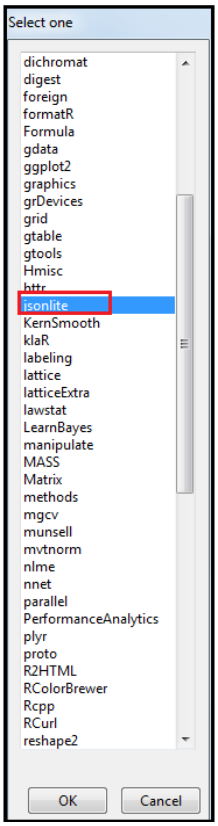
- Lightweight data storage
- Common format for data application programming interfaces (APIs)
- Similar structure to XML but different syntax / format
- Data stored as
  - Numbers
  - Strings
  - Boolean (true or false)
  - Array (ordered, comma separated enclosed in square brackets []
  - Object (unordered, comma separated collection of key: value pairs in curly brackets {}

Access JSON file using R

```
>  library(jsonlite)
> main_dir<-  "D:/R/data"
> setwd(file.path(main_dir))
> jsonData<- fromJSON("http://api.geonames.org/citiesJSON?north=44.1&south=-9.9&east=-22.4&west=55.2&lang=de&username=demo "
> names(jsonData)
[1] "geonames"
> head(jsonData)
$geonames
         lng geonameId countrycode         name           fclName toponymName
1  -99.12766   3530597          MX Mexiko-Stadt city, village,... Mexico City
2  116.39723   1816670          CN       Peking city, village,...     Beijing
3  120.98220   1701668          PH       Manila city, village,...      Manila
4   90.40744   1185241          BD        Dhaka city, village,...       Dhaka
5  126.97840   1835848          KR        Seoul city, village,...       Seoul
6  106.84513   1642911          ID      Jakarta city, village,...     Jakarta
7  139.69171   1850147          JP        Tokio city, village,...       Tokyo
8  121.53185   1668341          TW       Taipeh city, village,...      Taipei
9  -74.08175   3688689          CO      BogotÃ¡ city, village,...     BogotÃ¡
10 114.15769   1819729          HK    Hong Kong city, village,...   Hong Kong
                      fcodeName                       wikipedia       lat fcl population fcode
1  capital of a political entity en.wikipedia.org/wiki/Mexico_City 19.428472   P   12294193  PPLC
2  capital of a political entity     en.wikipedia.org/wiki/Beijing 39.907498   P   11716620  PPLC
3  capital of a political entity      en.wikipedia.org/wiki/Manila 14.604200   P   10444527  PPLC
4  capital of a political entity       en.wikipedia.org/wiki/Dhaka 23.710396   P   10356500  PPLC
5  capital of a political entity       en.wikipedia.org/wiki/Seoul 37.566000   P   10349312  PPLC
6  capital of a political entity     en.wikipedia.org/wiki/Jakarta -6.214623   P    8540121  PPLC
7  capital of a political entity       de.wikipedia.org/wiki/Tokyo 35.689500   P    8336599  PPLC
8  capital of a political entity      de.wikipedia.org/wiki/Taipei 25.047763   P    7871900  PPLC
9  capital of a political entity en.wikipedia.org/wiki/Bogot%C3%A1  4.609706   P    7674366  PPLC
10 capital of a political entity   en.wikipedia.org/wiki/Hong_Kong 22.285523   P    7012738  PPLC
```

Install jsonlite package as shown below:





5. Database mysql files

Assume we have a database test and created a table pet as shown below:

```
C:\Program Files\MySQL\MySQL Server 5.0\bin\mysql.exe
Enter password: ***
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4
Server version: 5.0.41-community-nt MySQL Community Edition (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| test               |
+--------------------+
3 rows in set (0.19 sec)

mysql> USE test
Database changed
mysql> CREATE TABLE pet (name VARCHAR(20),
    -> owner VARCHAR(20),
    -> species VARCHAR(20),
    -> sex CHAR(1),
    -> birth DATE,
    -> death DATE);
Query OK, 0 rows affected (0.97 sec)

mysql> SHOW tables;
+----------------+
| Tables_in_test |
+----------------+
| pet            |
+----------------+
1 row in set (0.00 sec)

mysql> desc pet;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| name    | varchar(20) | YES  |     | NULL    |       |
| owner   | varchar(20) | YES  |     | NULL    |       |
| species | varchar(20) | YES  |     | NULL    |       |
| sex     | char(1)     | YES  |     | NULL    |       |
| birth   | date        | YES  |     | NULL    |       |
| death   | date        | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
6 rows in set (0.18 sec)

mysql>
```

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| test               |
+--------------------+
4 rows in set (0.00 sec)

mysql> use test;
Database changed
mysql> select * from pet;
+--------+-------+---------+------+-------+-------+
| name   | owner | species | sex  | birth | death |
+--------+-------+---------+------+-------+-------+
| Caesar | Swamy | dog     | M    | NULL  | NULL  |
+--------+-------+---------+------+-------+-------+
1 row in set (0.00 sec)

mysql> _
```

5.1. RMySQL

- RMySQL is a database interface and MySQL driver for R.
- This version complies with the database interface definition as implemented in the package DBI 0.2-2.
- Download the latest version from
  http://cran.r-project.org/web/packages/RMySQL/index.html

5.2. Installing the RMySQL source package

1. Download Rtools from ***http://murdoch-sutherland.com/Rtools***, making sure to install the correct version for your R version.
2. Install a MySQL client library from ***http://www.mysql.com*** or ***http://dev.mysql.com.*** If you have already installed a MySQL server, you may want to re-run the install to ensure that you also installed client header and library files. Note that Xampp doesn't invoke the compilers.
3. Edit or create the file **Renviron.site** and add the variable **MYSQL_HOME** which contains the location of your MySQL install. The file typically isn't created when installing R, so you may need to create it yourself. You will want to place it under the **/etc** directory in your R Home area. If you don't where that is, you can issue **R.home()** at your R prompt. You will be adding a variable named **MYSQL_HOME** in the variable=value syntax. Here is an example:

   Location of Renviron.site: *C:/Program Files/R/R-3.1.1/etc/Renviron.site*

   Content is ***MYSQL_HOME = C:/Program Files/MySQL/MySQL Server 5.6***

4. Create the directory "***opt***" within *"C:\Program Files\MySQL\MySQL Server 5.6\lib"*
   - copy-paste the files as shown below:
     - *"C:\Program Files\MySQL\MySQL Server 5.6\lib\libmysql.lib" to "C:\Program Files\MySQL\MySQL Server 5.6\lib\opt\"*
     - *"C:\Program Files\MySQL\MySQL Server 5.6\lib\libmysql.dll" to "C:\Program Files\R\R-3.1.1\bin\i386\".*
5. Restart R and execute ***install.packages("RMySQL",type='source")***

Note:

- Issue Sys.getenv("MYSQL_HOME) from the R prompt. If it is empty, please re-check your Renviron.site file and place it in the correct directory.
- Binary versions of RMySQL is not supported.

Sample 1: Access data in mysql database using R

Untitled - R Editor

```
# ==============================================================================
# To fetch data from mysql database
# ==============================================================================
library("RMySQL")
# ==============================================================================
myDb        <- dbConnect(MySQL(), user="root",host="localhost",dbname="test")
on.exit(dbDisconnect(myDb))
# ==============================================================================
result      <- dbGetQuery(myDb,"show tables;")
rs          <- dbSendQuery(myDb,"select * from pet limit 10;")
data        <- fetch(rs, n=10)
print(result);print(data)
dbClearResult(rs)
# ==============================================================================
```

R Console

```
> # ==============================================================================
> # To fetch data from mysql database
> # ==============================================================================
> library("RMySQL")
> # ==============================================================================
> myDb<- dbConnect(MySQL(), user="root",host="localhost",dbname="test")
> on.exit(dbDisconnect(myDb))
> # ==============================================================================
> result<- dbGetQuery(myDb,"show tables;")
> rs<- dbSendQuery(myDb,"select * from pet limit 10;")
> data<- fetch(rs, n=10)
> print(result);print(data)
  Tables_in_test
1            pet
    name owner species sex birth death
1 Caesar Swamy     dog   M  <NA>  <NA>
> dbClearResult(rs)
[1] TRUE
> # ==============================================================================
```

Sample 2: Access data in mysql database using R

```
> library("RMySQL")
> # ==================================================================
> # Connecting to database and list tables
> # ==================================================================
> h1<- dbConnect(MySQL(), user="root",db="test",host="localhost")
> #
> # ==================================================================
> # List all tables within a database
> # ==================================================================
> #
> allTables <- dbListTables(h1) # List all tables in the data base
> print(allTables)
[1] "pet"
> # ==================================================================
> length(allTables)  # find total number of tables in the data base
[1] 1
```

```
> #
> # ==================================================================
> # Reading from a table
> # ==================================================================
> #
> data1<-dbReadTable(h1,"pet")
> head(data1)
    name owner species sex      birth death
1 Caesar Swamy     dog   M 2013-10-01  <NA>
2  Tommy Swamy     dog   M 2012-08-20  <NA>
3  Jimmy Swamy     dog   M 2005-08-10  <NA>
4  Tiger Swamy     dog   M 2006-02-12  <NA>
5  Nimmy Swamy     dog   F 2008-01-13  <NA>
6  Julio Swamy     dog   F 2009-03-04  <NA>
```

```
> #
> # ==================================================================
> # Get fields of a specific table
> # ==================================================================
> #
> dbListFields(h1,"pet")
[1] "name"    "owner"   "species" "sex"     "birth"   "death"
```

```
> #
> # ==================================================================
> dbGetQuery(h1,"select count(*) from pet")
  count(*)
1        7
```

```
> #
> # ==================================================================
> # Reading from a table
> # ==================================================================
> #
> data1<-dbReadTable(h1,"pet")
> head(data1)
    name owner species sex      birth death
1 Caesar Swamy     dog   M 2013-10-01  <NA>
2  Tommy Swamy     dog   M 2012-08-20  <NA>
3  Jimmy Swamy     dog   M 2005-08-10  <NA>
4  Tiger Swamy     dog   M 2006-02-12  <NA>
5  Nimmy Swamy     dog   F 2008-01-13  <NA>
6  Julio Swamy     dog   F 2009-03-04  <NA>
```

```
> #
> # ==================================================================
> # Select a sub-set of data
> # ==================================================================
> #
> qry<-dbSendQuery(h1,"select * from pet where  birth between
+  '2001-01-01' and curdate()")
> #
> data2few= fetch(qry,n=5)# select only upto 5 records
> print(data2few)
    name owner species sex      birth death
1 Caesar Swamy     dog   M 2013-10-01  <NA>
2  Tommy Swamy     dog   M 2012-08-20  <NA>
3  Jimmy Swamy     dog   M 2005-08-10  <NA>
4  Tiger Swamy     dog   M 2006-02-12  <NA>
5  Nimmy Swamy     dog   F 2008-01-13  <NA>
> dbClearResult(qry)
[1] TRUE
> #
> dim(data2few)
[1] 5 6
> #
> dbDisconnect(h1)
[1] TRUE
```

6. Hierarchical Data Format (HDF)

Ref: *http://en.wikipedia.org/wiki/Hierarchical_Data_Format*

- **Hierarchical Data Format** (**HDF**) is a set of file formats (**HDF4**, **HDF5**) designed to store and organize large amounts of numerical data. Originally developed at the National Center for Supercomputing Applications, it is supported by the non-profit HDF Group, whose mission is to ensure continued development of HDF5 technologies, and the continued accessibility of data stored in HDF.
- In keeping with this goal, the HDF format, libraries and associated tools are available under a liberal, BSD-like license for general use. HDF is supported by many commercial and non-commercial software platforms, including Java, MATLAB/Scilab, Octave, Interactive Data Language (IDL), Python, and R. The freely available HDF distribution consists of the library, command-line utilities, test suite source, Java interface, and the Java-based HDF Viewer (HDFView).
- For R interface for HDF format, refer to *http://www.bioconductor.org/packages/release/bioc/vignettes/rhdf5/inst/doc/rhdf5.pdf*
- Run the following commands from the R command shell to install the bioconductor package rhdf5.

6.1. Installation of the HDF5 package

- The package rhdf5 is an R interface for HDF5.

```
> source("http://bioconductor.org/biocLite.R")
trying URL 'http://www.bioconductor.org/packages/2.14/bioc/bin/windows/contrib/3.1/BiocInstaller_1.14.3.zip'
Content type 'application/zip' length 54023 bytes (52 Kb)
opened URL
downloaded 52 Kb


The downloaded binary packages are in
        C:\Users\PVS\AppData\Local\Temp\RtmpqkEqa2\downloaded_packages
Bioconductor version 2.14 (BiocInstaller 1.14.3), ?biocLite for help
A newer version of Bioconductor is available for this version of R,
  ?BiocUpgrade for help
> biocLite("rhdf5")
BioC_mirror: http://bioconductor.org
Using Bioconductor version 2.14 (BiocInstaller 1.14.3), R version 3.1.0.
Installing package(s) 'rhdf5'
also installing the dependency 'zlibbioc'

trying URL 'http://bioconductor.org/packages/2.14/bioc/bin/windows/contrib/3.1/zlibbioc_1.10.0.zip'
Content type 'application/zip' length 493535 bytes (481 Kb)
opened URL
downloaded 481 Kb

trying URL 'http://bioconductor.org/packages/2.14/bioc/bin/windows/contrib/3.1/rhdf5_2.8.0.zip'
Content type 'application/zip' length 5382708 bytes (5.1 Mb)
opened URL
downloaded 5.1 Mb

package 'zlibbioc' successfully unpacked and MD5 sums checked
package 'rhdf5' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\PVS\AppData\Local\Temp\RtmpqkEqa2\downloaded_packages
```

```
> #
> #==============================================
> # Createing an HDF5 file and group hierarchy
> #==============================================
> library(rhdf5)
> created = h5createFile("myexample.h5")
> created
[1] TRUE
> #==============================================
```

6.2. Read and write to a HDF5 file

- The HDF5 file can contain a group hierarchy. We create a number of groups and list the file content afterwards.

```
> # ==============================================================
> # Create a group hierarchy and list the file content afterwards
> # ==============================================================
> created = h5createGroup("myexample.h5","foo")
> created = h5createGroup("myexample.h5","baa")
> created = h5createGroup("myexample.h5","foo/foobaa")
> h5ls("myexample.h5")
  group   name     otype dclass dim
0     /    baa H5I_GROUP
1     /    foo H5I_GROUP
2  /foo foobaa H5I_GROUP
> # ==============================================================
```

- Objects can be written to the HDF5 file. Attributes attached to an object are written as well, if ***write.attributes = TRUE*** is given as argument to ***h5write***. *Note that not all R-attributes can be written as HDF5 as well.*

```
> # ==============================================================
> #
> # ==============================================================
> # Write objects to HDF5 file
> # ==============================================================
> A =matrix(1:10,nr=5,nc=2)
> h5write(A,"myexample.h5","foo/A")
> B = array(seq(0.1,2.0,by=0.1),dim=c(5,2,2))
> attr(B,"scale") <- "liter"
> h5write(B,"myexample.h5","foo/foobaa/B")
> h5ls("myexample.h5")
        group   name       otype  dclass       dim
0           /    baa   H5I_GROUP
1           /    foo   H5I_GROUP
2        /foo      A H5I_DATASET INTEGER     5 x 2
3        /foo foobaa   H5I_GROUP
4 /foo/foobaa      B H5I_DATASET   FLOAT 5 x 2 x 2
> #
> df = data.frame(1L:5L,seq(0,1,length.out=5),
+ c("ab","cde","fghi","a","s"), stringAsFactors=FALSE)
> h5write(df,"myexample.h5","df")
> h5ls("myexample.h5")
        group   name       otype   dclass       dim
0           /    baa   H5I_GROUP
1           /     df H5I_DATASET COMPOUND         5
2           /    foo   H5I_GROUP
3        /foo      A H5I_DATASET  INTEGER     5 x 2
4        /foo foobaa   H5I_GROUP
5 /foo/foobaa      B H5I_DATASET    FLOAT 5 x 2 x 2
> # ==============================================================
```

```
> # ==================================================================
> # Read HDF5 file
> # ==================================================================
> #
> # read data
> #
> readA = h5read("myexample.h5","foo/A")
> readB = h5read("myexample.h5","foo/foobaa/B")
> readdf = h5read("myexample.h5","df")
> readA
     [,1] [,2]
[1,]    1    6
[2,]    2    7
[3,]    3    8
[4,]    4    9
[5,]    5   10
> #
> # write and reading chuncks
> #
> h5write(c(12,13,14),"myexample.h5","foo/A",index=list(1:3,1))
> h5read("myexample.h5","foo/A")
     [,1] [,2]
[1,]   12    6
[2,]   13    7
[3,]   14    8
[4,]    4    9
[5,]    5   10
> # ==================================================================
```

- The function *h5dump* is similar to the function h5ls. If used with the argument ***load = FALSE***
  It produces the same result as h5ls, but with the group structure resolved as a hierarchy of lists.
  If the default argument ***load = TRUE*** is used all datasets from the HDF5 file are read.

```
> h5dump("myexample.h5",load=TRUE)
$baa
NULL

$df
  X1L.5L seq.0..1..length.out...5. c..ab....cde....fghi....a....s..
1      1                      0.00                               2
2      2                      0.25                               3
3      3                      0.50                               4
4      4                      0.75                               1
5      5                      1.00                               5

$foo
$foo$A
     [,1] [,2]
[1,]   12    6
[2,]   13    7
[3,]   14    8
[4,]    4    9
[5,]    5   10

$foo$foobaa
$foo$foobaa$B
, , 1

     [,1] [,2]
[1,]  0.1  0.6
[2,]  0.2  0.7
[3,]  0.3  0.8
[4,]  0.4  0.9
[5,]  0.5  1.0

, , 2

     [,1] [,2]
[1,]  1.1  1.6
[2,]  1.2  1.7
[3,]  1.3  1.8
[4,]  1.4  1.9
[5,]  1.5  2.0
```

7. Reading from the web

- You can read data directly from a web site without needing to download it to an intermediate file for import.
- Refer to *http://rconvert.com/sas-vs-r-code-compare/reading-data-from-url-sas-vs-r/*

```
> # ==============================================================================
> # Read data from a web site
> # ==============================================================================
> handle<-url("http://cran.r-project.org/doc/manuals/R-data.html#Overview-of-RDBMSs")
> dat1<-readLines(handle,10)  # Only 10 lines are read
> head(dat1,10)
 [1] "<!DOCTYPE html PUBLIC \"-//W3C//DTD HTML 4.01 Transitional//EN\" \"http://www.w3.org/TR/html4/loose.dtd\">"
 [2] "<html>"
 [3] "<!-- This manual is for R, version 3.1.1 (2014-07-10)."
 [4] ""
 [5] "Copyright (C) 2000-2013 R Core Team"
 [6] ""
 [7] "Permission is granted to make and distribute verbatim copies of this"
 [8] "manual provided the copyright notice and this permission notice are"
 [9] "preserved on all copies."
[10] ""
> # ==============================================================================
```

7.1. Parsing with XML

- For more details, please refer to *http://www.r-bloggers.com/r-and-the-web-for-beginners-part-ii-xml-in-r/*

```
> # ==============================================================================
> # Parsing data with XML
> # ==============================================================================
> require(XML)
> #
> # Use xmlTreeParse function to parse xml file directly from the web
> #
> data_xml <-xmlTreeParse("http://www.w3schools.com/xml/cd_catalog.xml")
> #
> # Access the top node by using the function xmlRoot
> #
> top_xml=xmlRoot(data_xml)
> #
> head(top_xml,1)
$CD
<CD>
 <TITLE>Empire Burlesque</TITLE>
 <ARTIST>Bob Dylan</ARTIST>
 <COUNTRY>USA</COUNTRY>
 <COMPANY>Columbia</COMPANY>
 <PRICE>10.90</PRICE>
 <YEAR>1985</YEAR>
</CD>

attr(,"class")
[1] "XMLNodeList"
> #
> # ==============================================================================
```

8. Direct interaction with files of various types

Refer to *http://127.0.0.1:14799/library/base/html/connections.html*

- In the R console, type ?connections to see details

Function	Details
file()	Open connection to a text file. The description is a path to the file to be opened or a complete URL.
url()	Open connection to a url. The description is a complete URL.
gzfile()	Open connection to a .gz file. The description is the path to a file compressed by gzip.
bzfile()	Open connection to ,bz2file. The description is the path to a file compressed by bzip2.
unz()	Reads single files within zip files, in binary mode. The description is the full path to the file, with '.zip'.