**R Exporting Data to CSV, Excel, SAS, STATA, and Text File**

**How to Export Data from R**

In this tutorial, we will learn how to export data from R environment to different formats.

To export data to the hard drive, you need the file path and an extension. First of all, the path is the location where the data will be stored. In this tutorial, you will see how to store data on:

* The hard drive
* Google Drive
* Dropbox

Secondly, R allows the users to export the data into different types of files. We cover the essential file's extension:

* csv
* xlsx
* RDS
* SAS
* SPSS
* STATA

Overall, it is not difficult to export data from R.

In this tutorial, you will learn-

* [Export Excel file](https://www.guru99.com/r-exporting-data.html" \l "1)
* [Export to different software](https://www.guru99.com/r-exporting-data.html" \l "2)
* [Export SAS file](https://www.guru99.com/r-exporting-data.html" \l "3)
* [Export STATA file](https://www.guru99.com/r-exporting-data.html" \l "4)
* [Interact with the Cloud Services](https://www.guru99.com/r-exporting-data.html" \l "5)
* [Google Drive](https://www.guru99.com/r-exporting-data.html" \l "6)
* [Export to Dropbox](https://www.guru99.com/r-exporting-data.html" \l "7)

**Export to Hard drive**

To begin with, you can save the data directly into the working directory. The following code prints the path of your working directory:

directory <-getwd()

directory

**Output:**

## [1] "/Users/15\_Export\_to\_do"

By default, the file will be saved in the below path.

For Mac OS:

/Users/USERNAME/Downloads/

For Windows:

C:\Users\USERNAME\Documents\

You can, of course, set a different path. For instance, you can change the path to the download folder.

**Create data frame**

First of all, let's import the mtcars dataset and get the mean of mpg and disp grouped by gear.

library(dplyr)

df <-mtcars % > %

select(mpg, disp, gear) % > %

group\_by(gear) % > %

summarize(mean\_mpg = mean(mpg), mean\_disp = mean(disp))

df

**Output:**:

## # A tibble: 3 x 3

## gear mean\_mpg mean\_disp

## <dbl> <dbl> lt;dbl>

## 1 3 16.10667 326.3000

## 2 4 24.53333 123.0167

## 3 5 21.38000 202.4800

The table contains three rows and three columns. You can create a CSV file with the function write.csv().

**Export CSV**

The basic syntax is:

write.csv(df, path)

arguments

-df: Dataset to save. Need to be the same name of the data frame in the environment.

-path: A string. Set the destination path. Path + filename + extension i.e. "/Users/USERNAME/Downloads/mydata.csv" or the filename + extension if the folder is the same as the working directory

Example:

write.csv(df, "table\_car.csv")

Code Explanation

* write.csv(df, "table\_car.csv"): Create a CSV file in the hard drive:
  + df: name of the data frame in the environment
  + "table\_car.csv": Name the file table\_car and store it as csv

**Note**: You can use the function write.csv2() to separate the rows with a semicolon.

write.csv2(df, "table\_car.csv")

**Note**: For pedagogical purpose only, we created a function called open\_folder() to open the directory folder for you. You just need to run the code below and see where the csv file is stored. You should see a file names table\_car.csv.

# Run this code to create the function

open\_folder <-function(dir){

if (.Platform['OS.type'] == "windows"){

shell.exec(dir)

} else {

system(paste(Sys.getenv("R\_BROWSER"), dir))

}

}

# Call the function to open the folder

open\_folder(directory)

**Export to Excel file**

Export data to Excel is trivial for Windows users and trickier for Mac OS user. Both users will use the library xlsx to create an Excel file. The slight difference comes from the installation of the library. Indeed, the library xlsx uses Java to create the file. Java needs to be installed if not present in your machine.

**Windows users**

If you are a Windows user, you can install the library directly with conda:

conda install -c r r-xlsx

Once the library installed, you can use the function write.xlsx(). A new Excel workbook is created in the working directory

library(xlsx)

write.xlsx(df, "table\_car.xlsx")

If you are a Mac OS user, you need to follow these steps:

* Step 1: Install the latest version of Java
* Step 2: Install library rJava
* Step 3: Install library xlsx

**Step 1)**You could download Java from official Oracle site and install it.

You can go back to Rstudio and check which version of Java is installed.

system("java -version")

At the time of the tutorial, the latest version of Java is 9.0.4.

**Step 2)**You need to install rjava in R. We recommended you to install R and Rstudio with Anaconda. Anaconda manages the dependencies between libraries. In this sense, Anaconda will handle the intricacies of rJava installation.

First of all, you need to update conda and then install the [library](https://anaconda.org/r/r-rjava). You can copy and paste the next two lines of code in the terminal.

conda - conda update

conda install -c r r-rjava

Next, open rjava in Rstudio

library(rJava)

**Step 3)**Finally, it is time to install xlsx. Once again, you can use [conda](https://anaconda.org/r/r-xlsx) to do it:

conda install -c r r-xlsx

Just as the windows users, you can save data with the function write.xlsx()

library(xlsx)

**Output:**

## Loading required package: xlsxjars

write.xlsx(df, "table\_car.xlsx")

**Export to different software**

Exporting data to different software is as simple as importing them. The library "haven" provides a convenient way to export data to

* spss
* sas
* stata

First of all, import the library. If you don't have "haven", you can go [here](https://anaconda.org/conda-forge/r-haven) to install it.

library(haven)

**SPSS file**

Below is the code to export the data to SPSS software:

write\_sav(df, "table\_car.sav")

**Export SAS file**

Just as simple as spss, you can export to sas

write\_sas(df, "table\_car.sas7bdat")

**Export STATA file**

Finally, haven library allows writing .dta file.

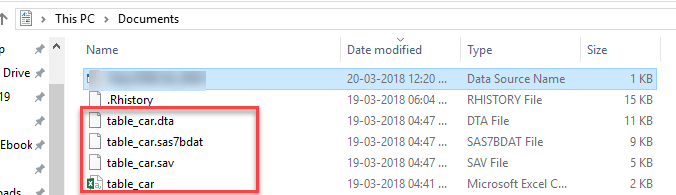
write\_dta(df, "table\_car.dta")

**R**

If you want to save a data frame or any other R object, you can use the save() function.

save(df, file ='table\_car.RData')

You can check the files created above in the present working directory

[](https://www.guru99.com/images/r_programming/032918_0502_RExportingD1.png)

**Interact with the Cloud Services**

Last but not least, R is equipped with fantastic libraries to interact with the cloud computing services. The last part of this tutorial deals with export/import files from:

* Google Drive
* Dropbox

**Note**: This part of the tutorial assumes you have an account with Google and Dropbox. If not, you can quickly create one for – Google Drive: <https://accounts.google.com/SignUp?hl=en> - Dropbox: <https://www.dropbox.com/h>

**Google Drive**

You need to install the library googledrive to access the function allowing to interact with Google Drive.

The library is not yet available at Anaconda. You can install it with the code below in the console.

install.packages("googledrive")

and you open the library.

library(googledrive)

For non-conda user, installing a library is easy, you can use the function install.packages('NAME OF PACKAGE) with the name of the package inside the parenthesis. Don't forget the ' '. Note that, R is supposed to install the package in the `libPaths() automatically. It is worth to see it in action.

**Upload to Google Drive**

To upload a file to Google drive, you need to use the function drive\_upload().

Each time you restart Rstudio, you will be prompted to allow access tidyverse to Google Drive.

The basic syntax of drive\_upload() is

drive\_upload(file, path = NULL, name = NULL)

arguments:

- file: Full name of the file to upload (i.e., including the extension)

- path: Location of the file- name: You can rename it as you wish. By default, it is the local name.

After you launch the code, you need to confirm several questions

drive\_upload%<("table\_car.csv", name ="table\_car")

**Output:**

## Local file:

## \* table\_car.csv

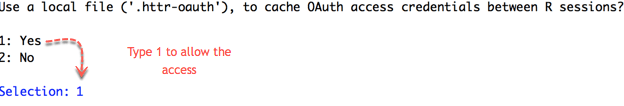
## uploaded into Drive file:

## \* table\_car: 1hwb57eT-9qSgDHt9CrVt5Ht7RHogQaMk

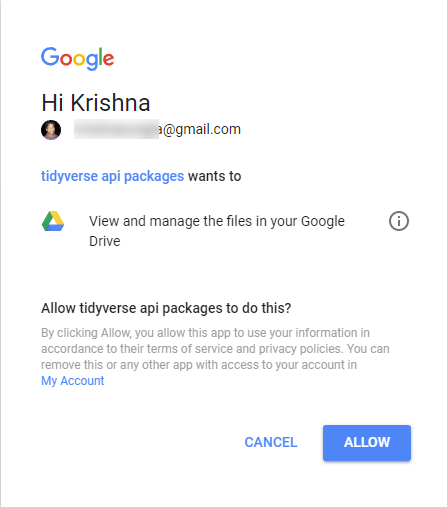
## with MIME type:

## \* text/csv

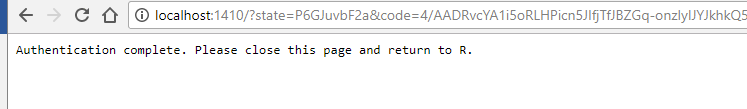
You type 1 in the console to confirm the access

[](https://www.guru99.com/images/r_programming/032918_0502_RExportingD2.png)

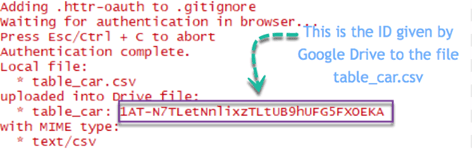
Then, you are redirected to Google API to allow the access. Click Allow.

[](https://www.guru99.com/images/r_programming/032918_0502_RExportingD3.png)

Once the authentication is complete, you can quit your browser.

[](https://www.guru99.com/images/r_programming/032918_0502_RExportingD4.png)

In the Rstudio's console, you can see the summary of the step done. Google successfully uploaded the file located locally on the Drive. Google assigned an ID to each file in the drive.

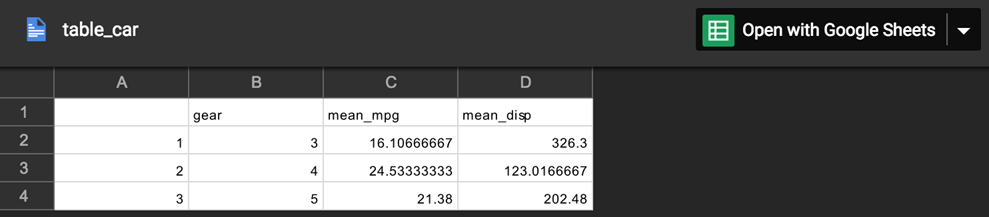
[](https://www.guru99.com/images/r_programming/032918_0502_RExportingD5.png)

You can see this file in Google Spreadsheet.

drive\_browse("table\_car")

**Output:**

You will be redirected to Google Spreadsheet

[](https://www.guru99.com/images/r_programming/032918_0502_RExportingD6.png)

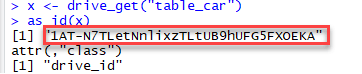
**Import from Google Drive**

Upload a file from Google Drive with the ID is convenient. If you know the file name, you can get its ID as follow:

**Note**: Depending on your internet connection and the size of your Drive, it takes times.

x <-drive\_get("table\_car")

as\_id(x)

[](https://www.guru99.com/images/r_programming/032918_0502_RExportingD7.png)

You stored the ID in the variable x. The function drive\_download() allows downloading a file from Google Drive.

The basic syntax is:

drive\_download(file, path = NULL, overwrite = FALSE)

arguments:

- file: Name or id of the file to download

-path: Location to download the file. By default, it is downloaded to the working directory and the name as in Google Drive

-overwrite = FALSE: If the file already exists, don't overwrite it. If set to TRUE, the old file is erased and replaced by the new one.

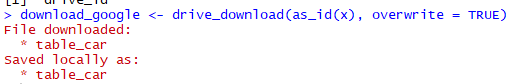
You can finally download the file:

download\_google & lt; - drive\_download(as\_id(x), overwrite = TRUE)

Code Explanation

* drive\_download(): Function to download a file from Google Drive
* as\_id(x): Use the ID to browse the file in Google Drive
* overwrite = TRUE: If file exists, overwrite it, else execution halted To see the name of the file locally, you can use:

**Output:**

[](https://www.guru99.com/images/r_programming/032918_0502_RExportingD8.png)

The file is stored in your working directory. Remember, you need to add the extenstion of the file to open it in R. You can create the full name with the function paste() (i.e. table\_car.csv)

google\_file <-download\_google$local\_path

google\_file

path <-paste(google\_file, ".csv", sep = "")

google\_table\_car <-read.csv(path)

google\_table\_car

**Output:**

## X gear mean\_mpg mean\_disp

## 1 1 3 16.10667 326.3000

## 2 2 4 24.53333 123.0167

## 3 3 5 21.38000 202.4800

Finally, you can remove the file from your Google drive.

## remove file

drive\_find("table\_car") %>%drive\_rm()

**Output:**

It's a slow process. Takes time to delete

**Export to Dropbox**

R interacts with Dropbox via the rdrop2 library. The library is not available at Anaconda as well. You can install it via the console

install.packages('rdrop2')

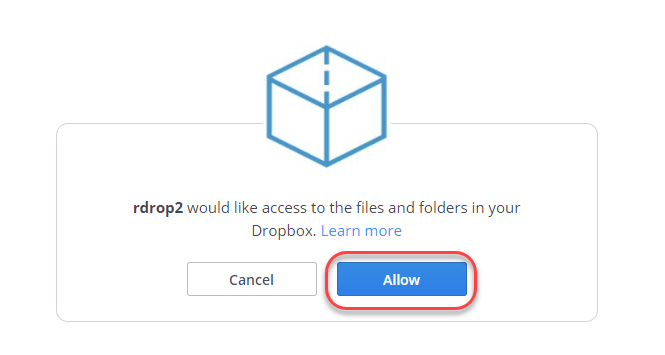
library(rdrop2)

You need to provide temporary access to Dropbox with your credential. After the identification is done, R can create, remove upload and download to your Dropbox.

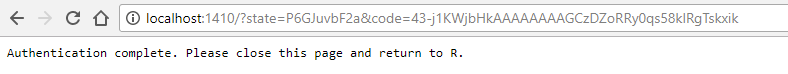
First of all, you need to give access to your account. The credentials are cached during all session.

drop\_auth()

You will be redirected to Dropbox to confirm the authentication.

[](https://www.guru99.com/images/r_programming/032918_0502_RExportingD10.png)

You will get a confirmation page. You can close it and return to R

[](https://www.guru99.com/images/r_programming/032918_0502_RExportingD11.png)

You can create a folder with the function drop\_create().

* drop\_create('my\_first\_drop'): Create a folder in the first branch of Dropbox
* drop\_create('First\_branch/my\_first\_drop'): Create a folder inside the existing First\_branch folder.

drop\_create('my\_first\_drop')

**Output:**

[https://www.guru99.com/images/r_programming/032918_0502_RExportingD12.png](https://www.guru99.com/images/r_programming/032918_0502_RExportingD12.png)

In DropBox

To upload the .csv file into your Dropbox, use the function drop\_upload().

Basic syntax:

drop\_upload(file, path = NULL, mode = "overwrite")

arguments:

- file: local path

- path: Path on Dropbox

- mode = "overwrite": By default, overwrite an existing file. If set to `add`, the upload is not completed.

drop\_upload('table\_car.csv', path = "my\_first\_drop")

**Output:**

At DropBox

You can read the csv file from Dropbox with the function drop\_read\_csv()

dropbox\_table\_car <-drop\_read\_csv("my\_first\_drop/table\_car.csv")

dropbox\_table\_car

**Output:**

## X gear mean\_mpg mean\_disp

## 1 1 3 16.10667 326.3000

## 2 2 4 24.53333 123.0167

## 3 3 5 21.38000 202.4800

When you are done using the file and want to delete it. You need to write the path of the file in the function drop\_delete()

drop\_delete('my\_first\_drop/table\_car.csv')

**Output:**

It is also possible to delete a folder

drop\_delete('my\_first\_drop')

**Output:**

**Summary**

We can summarize all the functions in the table below

| **Library** | **Objective** | **Function** |
| --- | --- | --- |
| base | Export csv | write.csv() |
| xlsx | Export excel | write.xlsx() |
| haven | Export spss | write\_sav() |
| haven | Export sas | write\_sas() |
| haven | Export stata | write\_dta() |
| base | Export R | save() |
| googledrive | Upload Google Drive | drive\_upload() |
| googledrive | Open in Google Drive | drive\_browse() |
| googledrive | Retrieve file ID | drive\_get(as\_id()) |
| googledrive | Dowload from Google Drive | download\_google() |
| googledrive | Remove file from Google Drive | drive\_rm() |
| rdrop2 | Authentification | drop\_auth() |
| rdrop2 | Create a folder | drop\_create() |
| rdrop2 | Upload to Dropbox | drop\_upload() |
| rdrop2 | Read csv from Dropbox | drop\_read\_csv |
| rdrop2 | Delete file from Dropbox | drop\_delete() |