

# R Data Cleaning Cheat Sheet

---

This file is a cheat sheet for doing ETL by R

- created by Weber Huang

## Table of Contents

---

0. [Start and load data](#)
  1. [Missing value](#)
  2. [Duplicated data](#)
  3. [Split](#)
  4. [Remove spaces](#)
  5. [Append](#)
  6. [Sort & filter](#)
  7. [Combine](#)
  8. [Format](#)
  9. [Replace](#)
  10. [Export data](#)
  11. [Ggplot x reorder](#)
  12. [Geocode](#)
  13. [Data shaping\(pivot\)](#)
  14. [Detect](#)
  15. [Regex](#)
  16. [Difference from datetime](#)
  17. [Trouble shooting](#)
- 

## Start and load data

First, we have to change the sys language in order to read the file without any warning message.

And then, switching the directory before loading anything.

```
# Mandarin Chinese  
Sys.setlocale(category="LC_ALL", locale="cht")
```

```
# load packages
library(xlsx)
library(readxl)
library(stringi)
library(stringr)
library(dplyr)
library(tidyr)
library(magrittr)
library(splitstackshape)
library(data.table)

# getwd
getwd()
setwd()

# load data
file <- read.csv("", stringsAsFactors = FALSE, na.strings = c("NA", ""))
file <- xlsx::read.xlsx(file="PChome.xlsx",sheetIndex=1,encoding = "UTF-8")
file <- readxl::read_excel("file.xlsx","sheet name or index")
```

## Missing value

```
# missing value
# detect NA
sapply(file, function(x) {sum(is.na(x))})
# missing value tratment
# drop NA
file = file %>% na.omit()
```

[This article](#) might help to handling the missing value.

## Duplicated data

```
# duplicated data
file = file %>% distinct()
```

## Split

```
# split column
# a
file = str_split_fixed(file$col, " ", 2)
# b
file = file %>% separate(col, c("new_col","new_col_2"), "")
# c:if ya don't know how many columns will be generated
file <- cbind(splitstackshape::cSplit(file, 'column name', ','))
```

## Remove spaces

```
# remove spaces
file$col = file$col %>% str_trim()
```

## Append

```
# append
file$col = file$col %>% paste("what ya wanna paste", sep = " ")
```

## Sort & filter

```
# sort & filter
file = file %>% filter(col=="what ya wanna find out")
file = file %>% arrange(desc(col1),desc(col2))
```

## Combine

```
# combine
# merge() for dataset have same id
file = file %>% unite("new column name", c("col1", "col2"),sep="_",remove=F)
```

## Format

```
# format
# time e.g 2018/01/02 22:22:22, only extract to hour
file$col = as.POSIXct(file$col)
file$col = format(file$col, format='%Y-%m-%d %H')
file = file %>%
  separate(col, c("date","hour"), " ")
file$date = file$date %>% as.Date()
```

## Replace

```
# replace
# whole
file$col[file$col == "what value wanna replace"] <- "new value"
# specific words
file$col = gsub("word","new word",file$col)
# Add a column to recognize if there is a unit x in column.
file <- file %>% mutate(result = stri_detect_regex(value,x))
```

## Export data

```
# export data
write.csv(file, file = "name_of_the_file.csv", row.names = FALSE, na = "")
```

## Ggplot x reorder

```
# ggplot x reorder
file$col <- factor(file$col, levels =file$col[order(file$col2)])
```

## Geocode

```
# geocode
require(ggmap)
# load file
file <- read.csv(file = "", na.strings = c("", "NA"),stringsAsFactors=FALSE)
# data cleaning
# === google map api key
register_google(key = " ### token type at here ### ")
# geocoding
for(i in 1:nrow(file))
{
  # Print("Working...")
  if (!(is.na(file$col[i]))) {
    result <- geocode(file$col[i], output = "latlon", source = "google")
    file$lon[i] <- as.numeric(result[1])
    file$lat[i] <- as.numeric(result[2])
    # origAddress$geoAddress[i] <- as.character(result[3])
  } else{
    file$lon[i] <- NA
    file$lat[i] <- NA
  }
}
write.csv(file, file = "", row.names = FALSE, na = "")
```

## Data shaping(pivot)

```
data shaping
file <- data.table::melt(file, id=1:25)
# columns after index 25 will be shape as colname in 26 and column value in 27.
```

## Detect

```
# detect
# detect if there is a pattern in file$col, return true-false
stringi::stri_detect_regex(file$col,pattern)
# count how many value in column, return integer
stringr::str_count(file$col, value)
# detect by index, return value in the range
stringi::stri_sub(file$col, start, end)
# detect letter's location
gregexpr(pattern = 'x', file$col)
stringr::str_locate_all(pattern = 'x',file$col)
```

## Regex

[This article](#) might help to dealing with string issues.

## Difference from datetime

```
# difference from datetime (dplyr)
file$datetime = as.Date(file$datetime, format='%Y/%m/%d # original style')
file = file %>%
  group_by(category) %>% arrange(category,datetime) %>%
  mutate(Diff = value - lag(value)) # lag() is method that finds the "previous"
values in a vector.
# if u wanna find the "next" values, use lead()
```

## Trouble shooting

### load file

- Q1: Error in `{r}(col.names, unique = TRUE) : invalid multibyte string 1`
- A1: Re-saving file as **CSV (comma delimited)(\*.csv)** from **CSV UTF-8 (comma delimited) (\*.csv)**
- Q2: Warning message of `{r}input string 1 is invalid in this locale`
- A2: This is a default language problem, since R's default language is English, if u wanna change default language, for example change English to Mandarin Chinese, add a line `Sys.setlocale(category="LC_ALL",locale="cht")` at the top of the script. Since this way isn't permanent, make sure add the line to the script what u wanna deal.