A03

資料的輸入與輸出

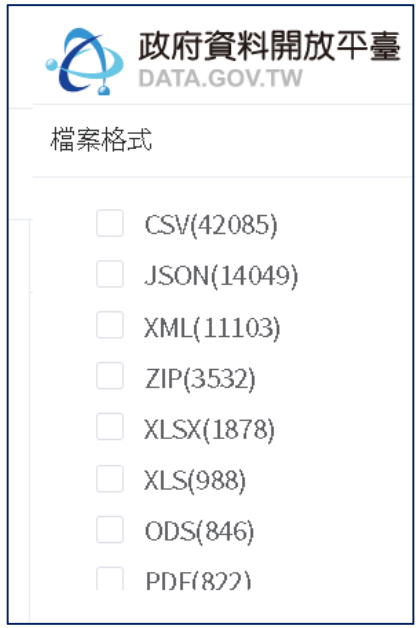
吳漢銘
國立政治大學 統計學系

http://www.hmwu.idv.tw

本章大綱&學習目標

- 標準輸入及輸出: **cat, sprint, print, scan, readline.**
- 讀取外部資料檔 (csv, xml, json, xls): **read.table, read.csv, read.delim, scan, xmlToDataFrame, fromJSON, read_excel.**

```
> read.table("input_testl.txt")
Error in file(file, "rt") : cannot open the connection
In addition: Warning message:
In file(file, "rt") :
  cannot open file 'input_testl.txt': No such file or directory
```

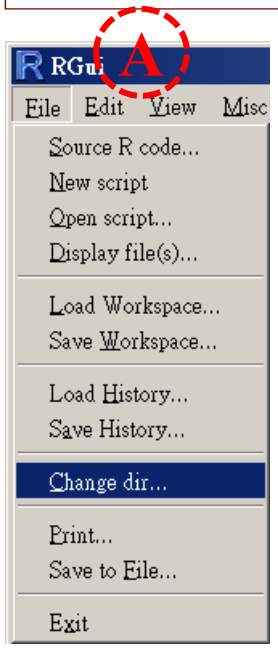


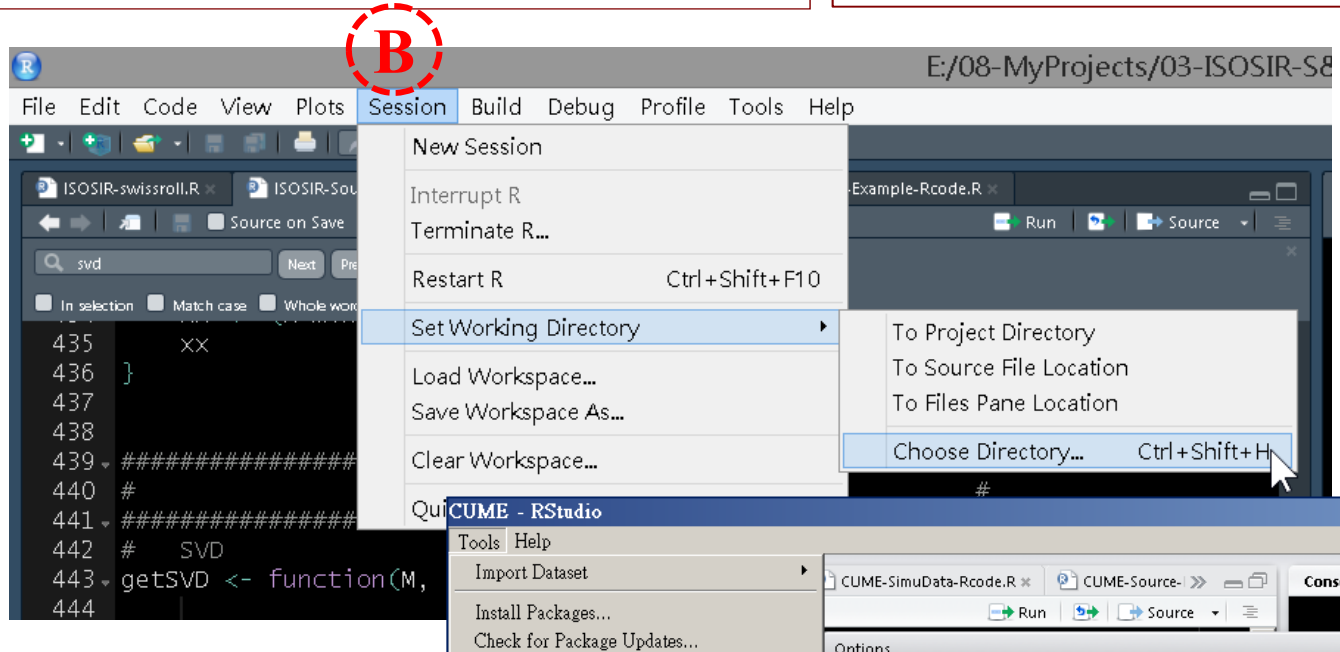
- 匯入內建資料、匯出資料(輸出)檔
- 使用 ODBC讀取Excel資料/讀取部份資料
- 讀取MySQL資料庫的資料: **RMySQL**
- R環境的記憶體設置、變數之標籤
- 資料中含有中文的編碼問題

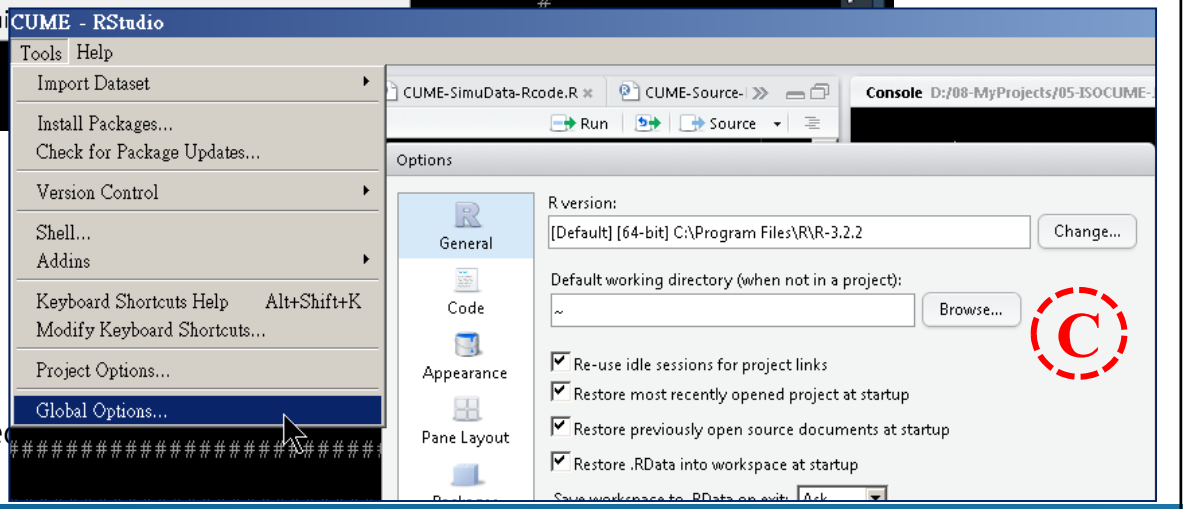
先確認或設定工作目錄 (建議 D 方法)

```
> getwd()
[1] "C:/Documents and Settings/user/My Documents"
> setwd("C:\\Program Files\\R\\working")
> getwd()
[1] "C:/Program Files/R/working"
```

```
> dir()
 [1] "Ccode.R"      "cellcycle"
 [3] "data"         "epstopdf.exe"
 [5] "fig"          "MWI2016-ts.pdf"
 [7] "myC"          "myCcode.c"
```







D

RStudio: 建立一工作專案，

並新增一資料目錄data。

(見Error in file(file, "rt") : cannot open the conne
cannot open file, No such file or directory)

cat {base}: Concatenate and Print

Description: Outputs the objects, concatenating the representations. cat performs much less conversion than print.

Usage:
```
cat(... , file = "", sep = " ", fill = FALSE, labels = NULL,
          append = FALSE)
```

```
> stdout()
description       class        mode        text      opened    can read
can write   "stdout"  "terminal"         "w"      "text"
"opened"        "no"       "yes"
> ?stdout()
>
> cat("Hello R users!\n")
Hello R users!
> a <- c(1,2,3)
> cat("Here is a list: ", a, "\n")
Here is a list:  1 2 3
>
> cat("3 + 5 =", 3+ 5, "\n" )
3 + 5 = 8
> cat("A test list: ", paste("Test", 1:3, sep="-"), "\n")
A test list:  Test-1 Test-2 Test-3
```

標準輸出，從營幕輸出: cat

C:\Program Files\R\working\Example2.R - R Editor

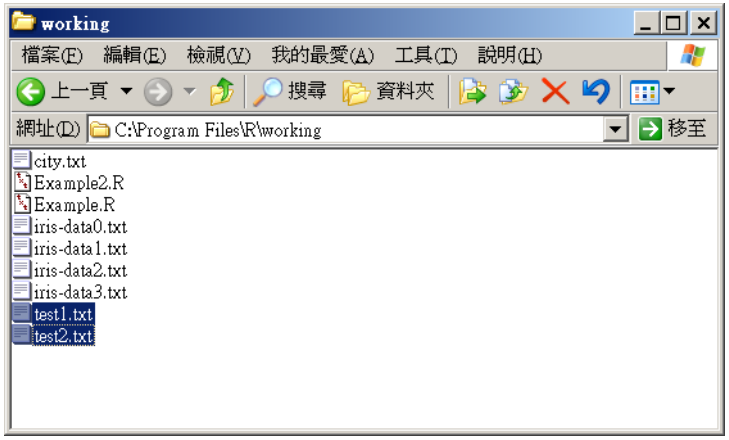
```
a1 <- 1.2123344
a2 <- 23.3
a3 <- 10/3

cat("iteration", "\t", "mathod-1","\t","method-2", "\t", "method-3\n")
for (i in 1:3){
    cat(i,"\t", round(a1, 3),"\t",  round(a2, 3), "\t", round(a3, 3),"\n")
    a1 <- a1+i
    a2 <- a2*i
    a3 <- a3/i
}
```

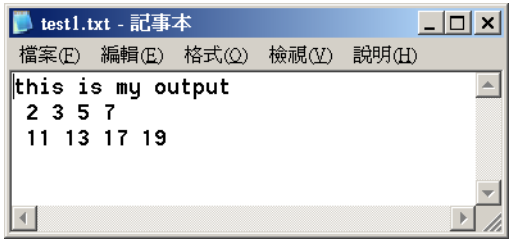
```
> a1 <- 1.2123344
> a2 <- 23.3
> a3 <- 10/3
>
> cat("iteration", "\t", "mathod-1","\t","method-2", "\t", "method-3\n")
iteration       mathod-1        method-2        method-3
> for (i in 1:3){
+     cat(i,"\t", round(a1, 3),"\t",  round(a2, 3), "\t", round(a3, 3),"\n")
+     a1 <- a1+i
+     a2 <- a2*i
+     a3 <- a3/i
+ }
1       1.212   23.3    3.333
2       2.212   23.3    3.333
3       4.212   46.6    1.667
>
```

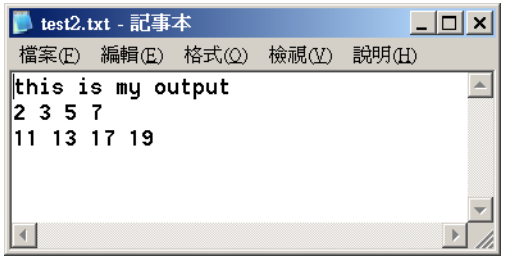
```
> source("Example2.R")
iteration method-1  method-2 method-3
1           1.212     23.3    3.333
2           2.212     23.3    3.333
3           4.212     46.6    1.667
```

標準輸出，輸出至檔案: cat

```
> cat("this is my output","\n", "2 3 5 7","\n", "11 13 17 19", file = "test1.txt")
> cat("this is my output", "2 3 5 7", "11 13 17 19", file = "test2.txt", sep = "\n")
```








```
> cat("today", "is", date(), sep="\t", "\n")
today   is      Wed Nov 08 00:14:43 2017
```

See also:

- print
- sprintf
- print.data.frame
- paste

sprintf {base}:
Use C-style String Formatting Commands

Description: A wrapper for the C function sprintf, that returns a character vector containing a formatted combination of text and variable values.

Usage: `sprintf(fmt, ...)`

```
> sprintf("%f", pi)
[1] "3.141593"
> sprintf("%.3f", pi)
[1] "3.142"
> sprintf("%1.0f", pi)
[1] "3"
> sprintf("%5.1f", pi)
[1] "  3.1"
> sprintf("%05.1f", pi)
[1] "003.1"
> sprintf("%+f", pi)
[1] "+3.141593"
> sprintf("% f", pi)
[1] " 3.141593"
> sprintf("%-10f", pi)# left justif
[1] "3.141593  "
> sprintf("%e", pi)
[1] "3.141593e+00"
> sprintf("%s is %f feet tall", "Sven", 7.1)
[1] "Sven is 7.100000 feet tall"
> sprintf("%.0f%% said yes (out of a sample of size %.0f)", 66.666, 3)
[1] "67% said yes (out of a sample of size 3)"
```

```
> pi
[1] 3.141593
```

- **d**: Integer value.
- **f**: Double precision value, in "fixed point" decimal notation
- **e**: Double precision value, in "exponential" decimal notation.
- **s**: Character string.
- **%m.n**: denoting the field width (**m**) and the precision (**n**).
- **%-**: Left adjustment of converted argument in its field.

```
> a <- c(0, 1, 12, 123)
> sprintf("name_%03d", a)
[1] "name_000" "name_001" "name_012" "name_123"
> paste("name", formatC(a, width=3, flag="0"), sep="_")
[1] "name_000" "name_001" "name_012" "name_123"
```

cat() 和 print()

```
> cat("hello")
hello> print("hello")
[1] "hello"
> class(cat("hello"))
hello[1] "NULL"
> class(print("hello"))
[1] "hello"
[1] "character"
>
> a <- cat("hello")
hello> b <- print("hello")
[1] "hello"
> class(a)
[1] "NULL"
> class(b)
[1] "character"
>
> cat("Today is: ", date(), "\n")
Today is:  Wed Nov 08 00:48:25 2017
> print("Today is: ", date())
Error in print.default("Today is: ", date()) : 'digits' 引數不正確
此外: Warning message:
In print.default("Today is: ", date()) : 強制變更過程中產生了 NA
>
> cat(head(iris, 2))
Error in cat(list(...), file, sep, fill, labels, append) :
  'cat' 目前還不能用 1 引數 (類型 'list')
> print(head(iris, 2))
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
```

```
> dice1 <- sample(1:6, 10, replace=TRUE)
> dice2 <- sample(1:6, 10, replace=TRUE)
> mytable <- table(dice1, dice2)
> mytable
     dice2
dice1 1 2 4 5 6
    1 1 0 1 0 0
    3 0 1 0 0 0
    4 0 0 0 0 1
    5 1 0 0 1 1
    6 1 0 0 1 1
> print(mytable, zero.print = ".")
     dice2
dice1 1 2 4 5 6
    1 1 . 1 . .
    3 . 1 . . .
    4 . . . . 1
    5 1 . . 1 1
    6 1 . . 1 1
```

cat is valid only for atomic types (logical, integer, real, complex, character) and names. (not on a non-empty list or any type of object.)
print is a generic function so you can define a specific implementation for a certain S3 class.

標準輸入 (Standard Input)
從營幕/鍵盤輸入: scan( )

```
> stdin()
description       class        mode        text      opened    can read    can write
    "stdin"  "terminal"         "r"      "text"    "opened"       "yes"    "no"
```

```
> a <- scan()
1: 1 2
3: 3
4:
Read 3 items
> a
[1] 1 2 3
> b <- scan(nmax=1)
1: 5
Read 1 item
> b
[1] 5
> b <- scan(nmax=1, quiet=TRUE)
1: 5
> b
[1] 5
```

logical, integer, numeric, complex, character, raw and list

```
> cc <- scan(what = "character", quiet = TRUE)
1: this is a test
5:
> cc
[1] "this" "is"   "a"    "test"
```

```
> cc <- scan(what = "character", quiet = TRUE)
1: "this is a test" "are you ok?"
3:
> cc
[1] "this is a test" "are you ok?"
```

標準輸入 (Standard Input)
從營幕/鍵盤輸入: scan(), readline()

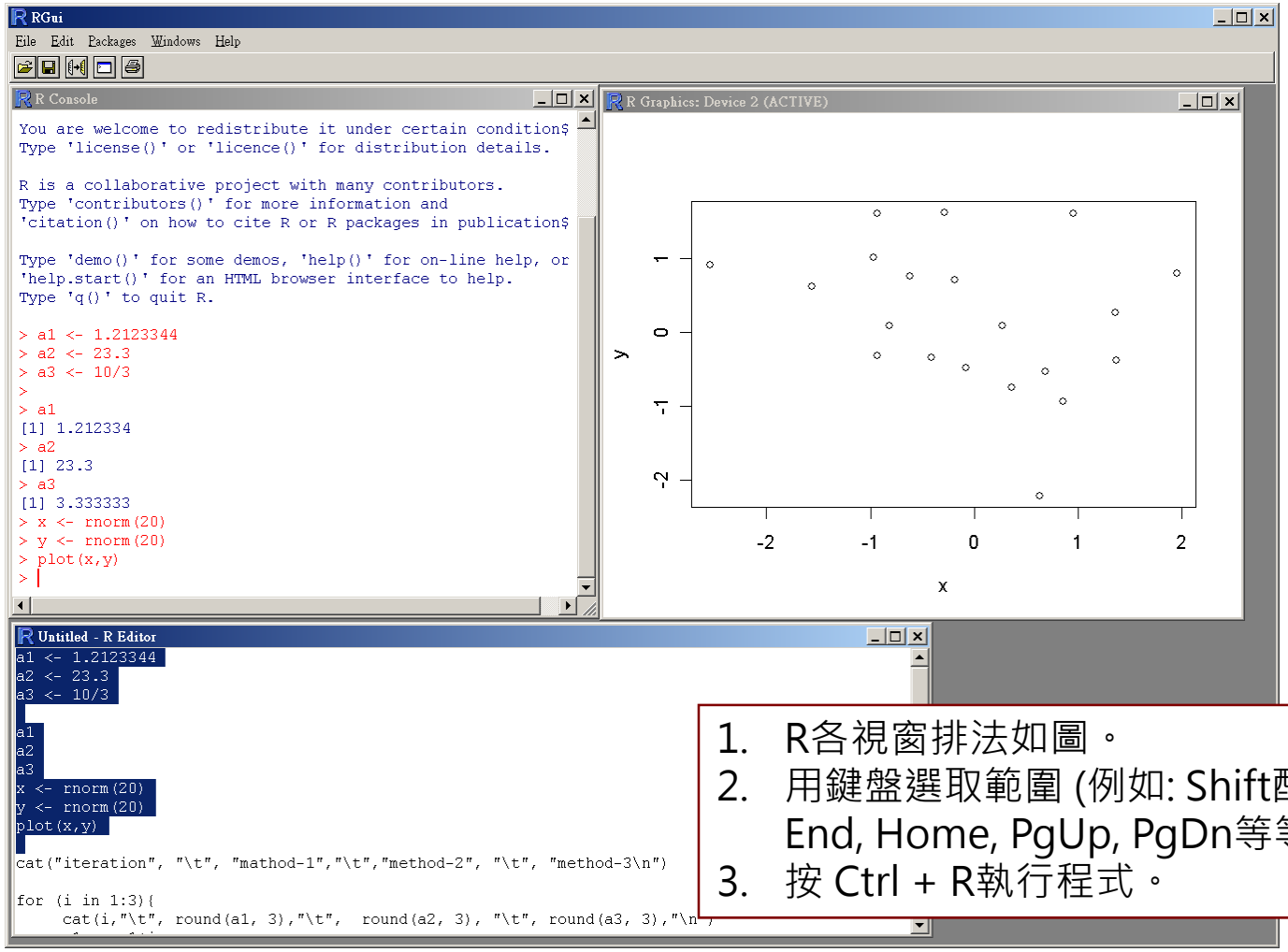
```
> d <- scan(what = list(name = "character", age = "numeric", isboy = "logical"))
1: john 28 true
2: mary 11 false
3:
Read 2 records
> d
$name
[1] "john" "mary"

$age
[1] "28" "11"

$isboy
[1] "true"  "false"
```

```
> e <- scan()
1: 1 2 3
4: 4 5 6
7: 7 8 9
10:
Read 9 items
> e.mat <- matrix(e, ncol = 3, byrow = TRUE)
> e.mat
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

```
> my.name <- readline("請輸入您的姓字: ")
請輸入您的姓字: 吳小銘
> my.age <- readline("請輸入您的年紀: ")
請輸入您的年紀: 25
> my.age <- as.numeric(my.age)
> cat("您輸入的資訊如下:\n", "姓名: ", my.name, ",
年紀: ", my.age, "\n")
您輸入的資訊如下:
 姓名:  吳小銘 , 年紀:  25
```

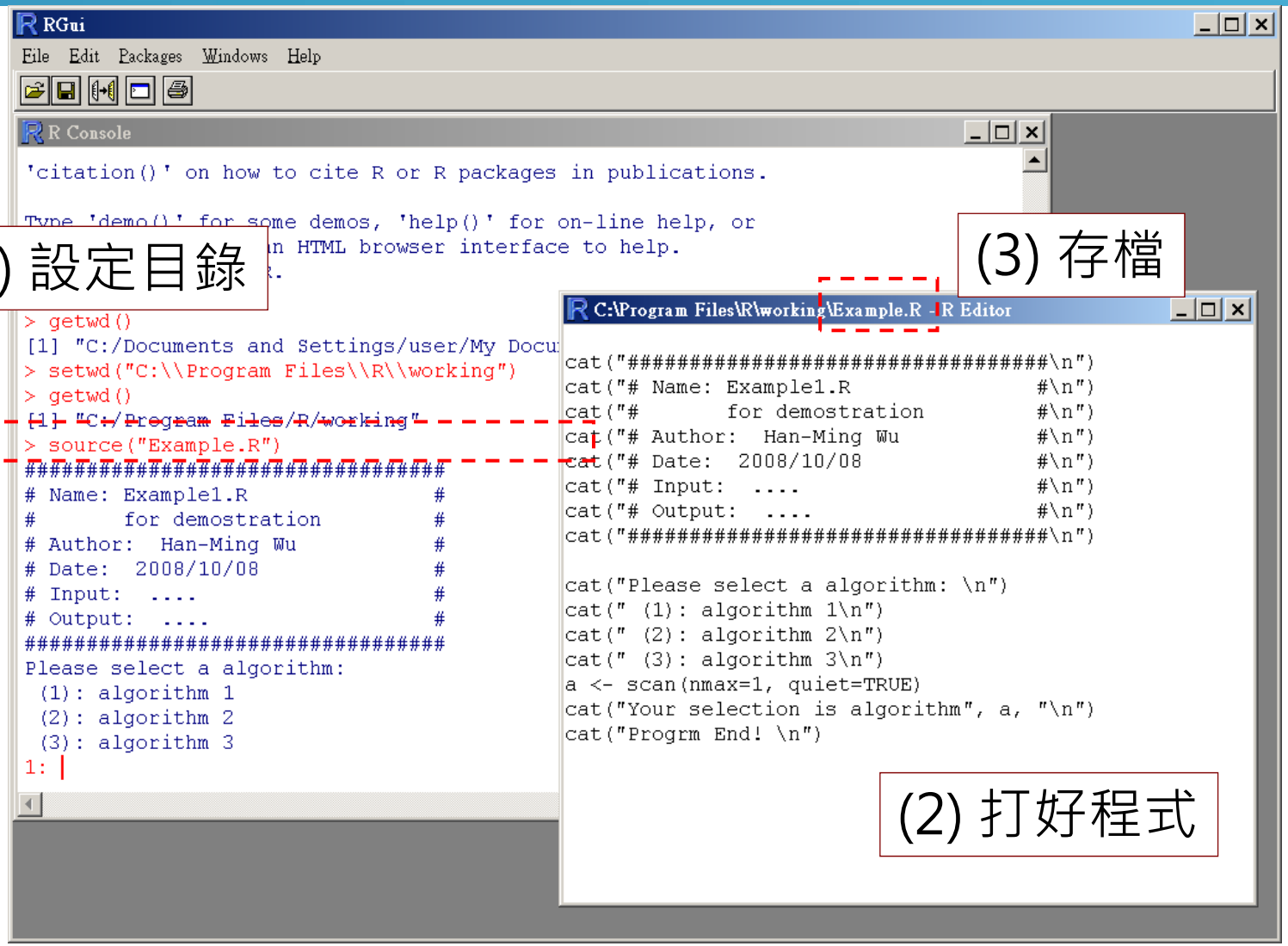
RGui 小技巧

建議使用RStudio建立一個工作專案(Project)，進行程式撰寫及執行。



1. R各視窗排法如圖。
2. 用鍵盤選取範圍 (例如: Shift配合上下左右鍵，End, Home, PgUp, PgDn等等)。
3. 按 Ctrl + R執行程式。

課堂練習



(1) 設定目錄

(3) 存檔

(4) 執行程式

(2) 打好程式

(5) 請用RStudio建立一個專案，並實作課堂練習1

讀取外部純文字表格資料檔: read.table()

```
Reads a file in table format and creates a data frame from it, with cases corresponding
to lines and variables to fields in the file.

read.table(file, header = FALSE, sep = "", quote = "\"'",
           dec = ".", row.names, col.names,
           as.is = !stringsAsFactors,
           na.strings = "NA", colClasses = NA, nrows = -1,
           skip = 0, check.names = TRUE, fill = !blank.lines.skip,
           strip.white = FALSE, blank.lines.skip = TRUE,
           comment.char = "#",
           allowEscapes = FALSE, flush = FALSE,
           stringsAsFactors = default.stringsAsFactors(),
           fileEncoding = "", encoding = "unknown", text, skipNul = FALSE)
```

read.table()

- read in a rectangular grid of data.
- 文字檔.txt, 以空白(" ")或Tab("\t")做區隔。
- **read.table()** is an inefficient way to read in very large numerical matrices. (use **scan()**)

read.csv()

- 格式檔.csv，以"," 做區隔

read.table() or **read.csv()** are almost identical.

讀取外部純文字表格資料檔: read.table()

- 注意資料是否有「欄位名稱」
- 分隔符號是什麼?

first line: a name for each variable
`header = TRUE`

row label

values

factors

iris-data1.txt

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa
11			1.5		
12			1.6		
13			1.4		
14			1.1		
15			1.2		
16	5.7	4.4	1.5	0.4	setosa
17	5.4	3.9	1.3	0.4	setosa
18	5.1	3.5	1.4	0.3	setosa
19	5.7	3.8	1.7	0.3	setosa
20	5.1	3.8	1.5	0.3	setosa
21	5.4	3.4	1.7	0.2	setosa
22	5.1	3.7	1.5	0.4	setosa
23	4.6	3.6	1	0.2	setosa

iris-data2.txt

no	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa
11	5.4	3.7	1.5	0.2	setosa
12	4.8	3.4	1.6	0.2	setosa
13	4.8	3	1.4	0.1	setosa
14	4.3	3	1.1	0.1	setosa
15	5.8	4	1.2	0.2	setosa
16	5.7	4.4	1.5	0.4	setosa
17	5.4	3.9	1.3	0.4	setosa
18	5.1	3.5	1.4	0.3	setosa
19	5.7	3.8	1.7	0.3	setosa
20	5.1	3.8	1.5	0.3	setosa
21	5.4	3.4	1.7	0.2	setosa
22	5.1	3.7	1.5	0.4	setosa
23	4.6	3.6	1	0.2	setosa

iris-data3.txt

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa
5.4	3.7	1.5	0.2	setosa
4.8	3.4	1.6	0.2	setosa
4.8	3	1.4	0.1	setosa
4.3	3	1.1	0.1	setosa
5.8	4	1.2	0.2	setosa
5.7	4.4	1.5	0.4	setosa
5.4	3.9	1.3	0.4	setosa
5.1	3.5	1.4	0.3	setosa
5.7	3.8	1.7	0.3	setosa
5.1	3.8	1.5	0.3	setosa
5.4	3.4	1.7	0.2	setosa
5.1	3.7	1.5	0.4	setosa
4.6	3.6	1	0.2	setosa

```
my.data <- read.table("iris-data1.txt")

my.data <- read.table("iris-data2.txt", header = TRUE, row.names = 1)

my.data <- read.table("iris-data3.txt", header = TRUE, sep = "\t")
```

課堂練習

```
>  my.data <- read.table("iris-data0.txt", header=FALSE)
>  dim(my.data)
[1] 150   5
>  my.data[1:3,]
   V1  V2  V3  V4     V5
1 5.1 3.5 1.4 0.2 setosa
2 4.9 3.0 1.4 0.2 setosa
3 4.7 3.2 1.3 0.2 setosa
>  attributes(my.data)
$names
[1] "V1" "V2" "V3" "V4" "V5"

$class
[1] "data.frame"

$row.names
  [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
...
[145] 145 146 147 148 149 150

> row.names(my.data)
  [1] "1"   "2"   "3"   "4"   "5"   "6"   "7"   "8"   "9"   "10"  "11"  "12"
..
[145] "145" "146" "147" "148" "149" "150"
> names(my.data)
[1] "V1" "V2" "V3" "V4" "V5"
> colnames(my.data)
[1] "V1" "V2" "V3" "V4" "V5"
```

```
> head(my.data)
> tail(my.data)
```

iris-data0.txt

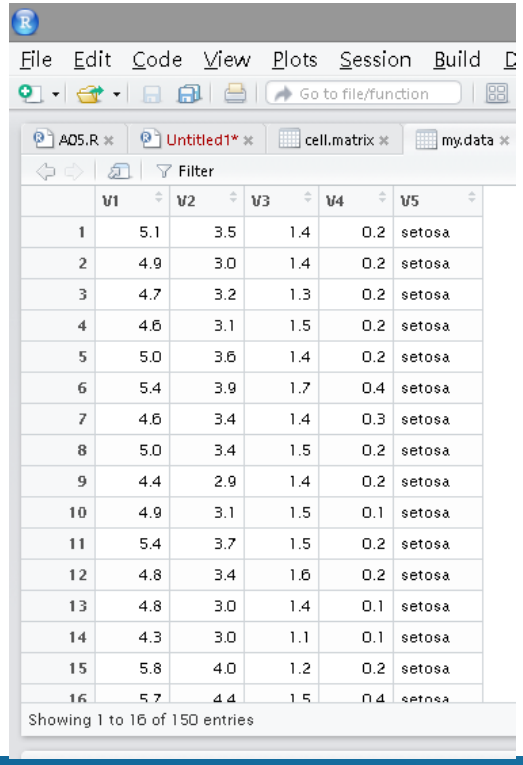
5.1	3.5	1.4	0.2	setosa
4.9	3	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa
5.4	3.7	1.5	0.2	setosa
4.8	3.4	1.6	0.2	setosa
4.8	3	1.4	0.1	setosa
4.3	3	1.1	0.1	setosa
5.8	4	1.2	0.2	setosa
5.7	4.4	1.5	0.4	setosa
5.4	3.9	1.3	0.4	setosa
5.1	3.5	1.4	0.3	setosa
5.7	3.8	1.7	0.3	setosa
5.1	3.8	1.5	0.3	setosa
5.4	3.4	1.7	0.2	setosa
5.1	3.7	1.5	0.4	setosa
4.6	3.6	1	0.2	setosa

課堂練習

```
> View(my.data)
> str(my.data)
'data.frame':   150 obs. of  5 variables:
 $ V1: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
 $ V2: num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 $ V3: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
 $ V4: num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
 $ V5: Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

iris-data0.txt

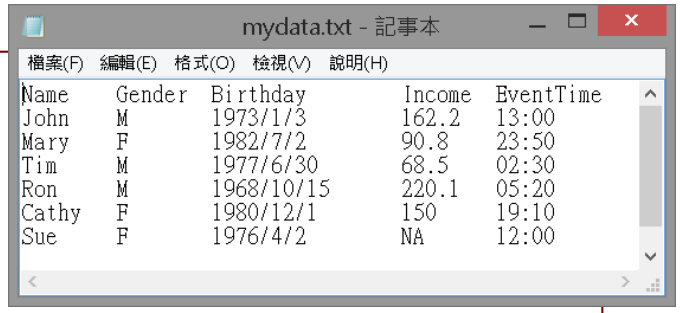
5.1	3.5	1.4	0.2	setosa
4.9	3	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa
5.4	3.7	1.5	0.2	setosa
4.8	3.4	1.6	0.2	setosa
4.8	3	1.4	0.1	setosa
4.3	3	1.1	0.1	setosa
5.8	4	1.2	0.2	setosa
5.7	4.4	1.5	0.4	setosa
5.4	3.9	1.3	0.4	setosa
5.1	3.5	1.4	0.3	setosa
5.7	3.8	1.7	0.3	setosa
5.1	3.8	1.5	0.3	setosa
5.4	3.4	1.7	0.2	setosa
5.1	3.7	1.5	0.4	setosa
4.6	3.6	1	0.2	setosa

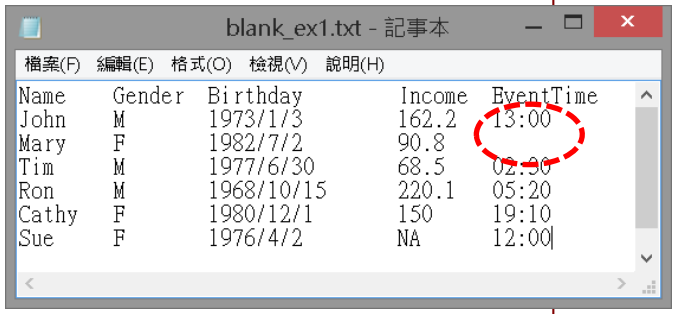


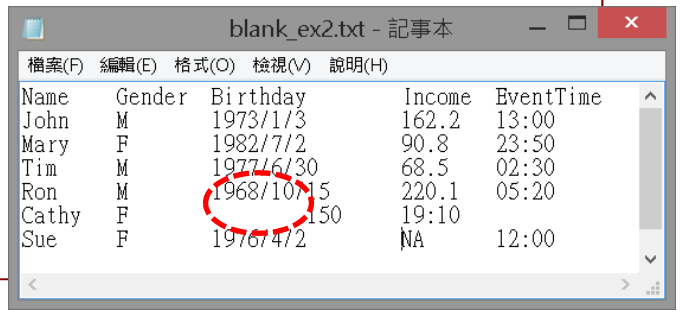
See also:
readr package

純文字表格資料含有空格"blank"

```
> x <- read.table("mydata.txt", header = T)
> head(x)
   Name Gender   Birthday Income EventTime
1  John      M   1973/1/3  162.2     13:00
...
6   Sue      F   1976/4/2     NA     12:00
> x.b1 <- read.table("blank_ex1.txt", header = T)
Error in scan(file = file, what = what, sep = sep, quote = quote, dec = dec,  :
  第 2 列沒有 5 個元素
> x.b1 <- read.table("blank_ex1.txt", header = T, fill = T)
> head(x.b1)
   Name Gender   Birthday Income EventTime
1  John      M   1973/1/3  162.2     13:00
2  Mary      F   1982/7/2   90.8
3   Tim      M  1977/6/30   68.5     02:30
...
6   Sue      F   1976/4/2     NA     12:00
> x.b2 <- read.table("blank_ex2.txt", header = T)
Error in scan(file = file, what = what, sep = sep, quote = quote, dec = dec,  :
  第 5 列沒有 5 個元素
> x.b2 <- read.table("blank_ex2.txt", header = T, fill = T)
> head(x.b2)
   Name Gender   Birthday Income EventTime
1  John      M   1973/1/3  162.2     13:00
...
5 Cathy      F        150  19:10
6   Sue      F   1976/4/2   <NA>     12:00
```



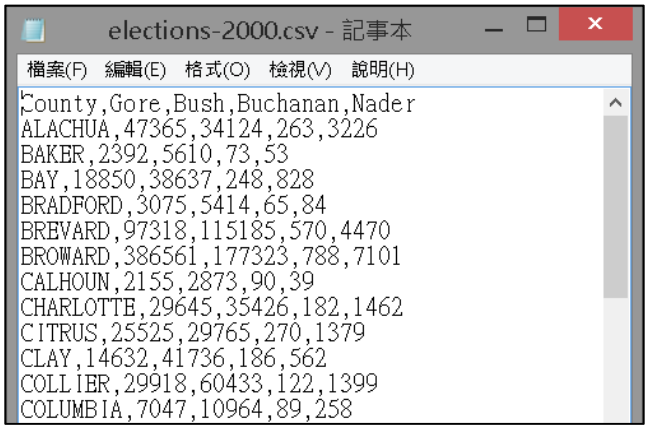


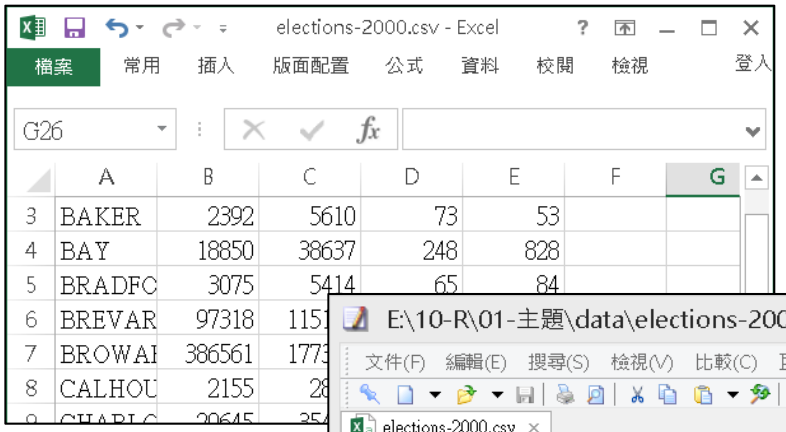


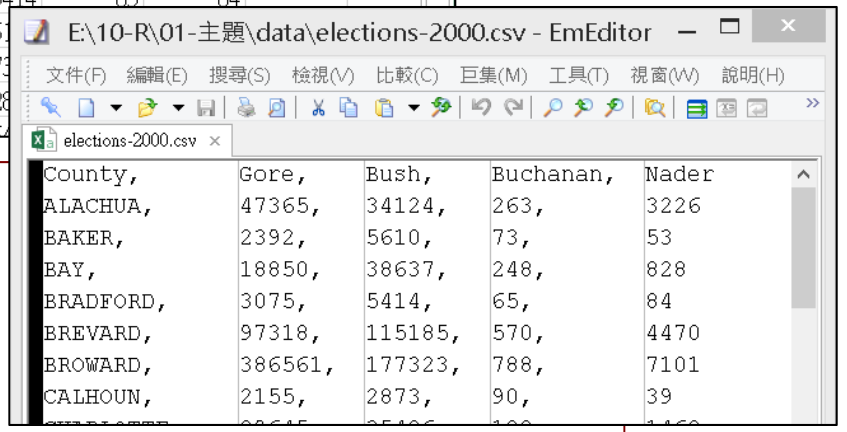
讀取 CSV檔 (逗點分隔值): read.csv( )

```
read.csv(file, header = TRUE, sep = ",", quote = "\"",
         dec = ".", fill = TRUE, comment.char = "", ...)

read.csv2(file, header = TRUE, sep = ";", quote = "\"",
          dec = ",", fill = TRUE, comment.char = "", ...)
```

fill: if TRUE then in case the rows have unequal length, blank fields are implicitly added.



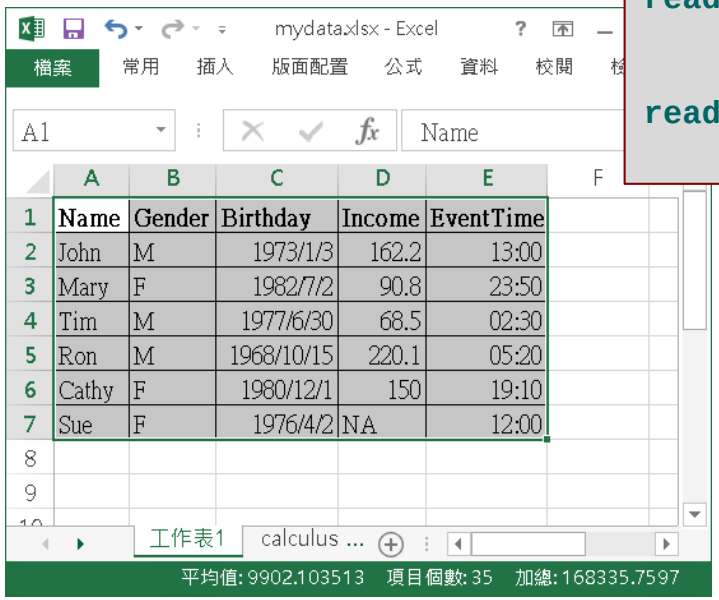





```
> elections <- read.csv("elections-2000.csv")
> head(elections)
    County   Gore   Bush Buchanan Nader
1  ALACHUA  47365  34124      263  3226
...
6  BROWARD 386561 177323      788  7101
> str(elections)
'data.frame':   67 obs. of  5 variables:
 $ County  : Factor w/ 67 levels "ALACHUA","BAKER",..: 1 2 3 4 5 6 7 8 9 10 ...
 $ Gore    : int  47365 2392 18850 3075 97318 386561 2155 29645 25525 14632 ...
 $ Bush    : int  34124 5610 38637 5414 115185 177323 2873 35426 29765 41736 ...
 $ Buchanan: int  263 73 248 65 570 788 90 182 270 186 ...
 $ Nader   : int  3226 53 828 84 4470 7101 39 1462 1379 562 ...
```

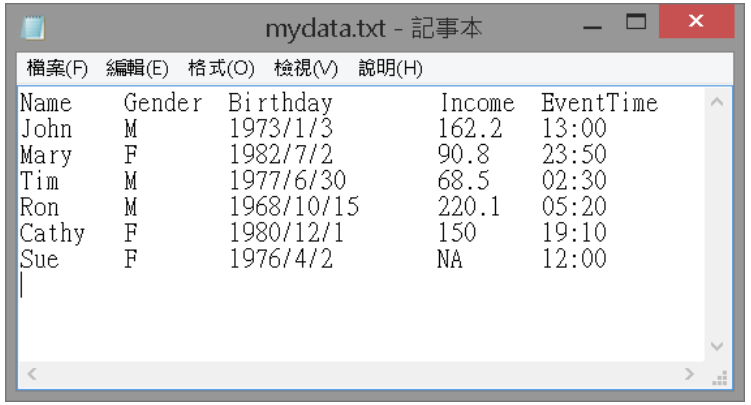
讀取"TAB"為分隔之資料檔: read.delim()

```
read.delim(file, header = TRUE, sep = "\t", quote = "\"",
           dec = ".", fill = TRUE, comment.char = "", ...)

read.delim2(file, header = TRUE, sep = "\t", quote = "\"",
            dec = ",", fill = TRUE, comment.char = "", ...)
```



	Name	Gender	Birthday	Income	EventTime
1	John	M	1973/1/3	162.2	13:00
2	Mary	F	1982/7/2	90.8	23:50
3	Tim	M	1977/6/30	68.5	02:30
4	Ron	M	1968/10/15	220.1	05:20
5	Cathy	F	1980/12/1	150	19:10
6	Sue	F	1976/4/2	NA	12:00

Ctrl + A
Ctrl + C
Ctrl + V
Ctrl + S




```
> mydata <- read.delim("mydata.txt")
> head(mydata)
   Name Gender   Birthday Income EventTime
1  John      M   1973/1/3  162.2     13:00
...
6   Sue      F   1976/4/2     NA     12:00
> str(mydata)
'data.frame':   6 obs. of  5 variables:
 $ Name     : Factor w/ 6 levels "Cathy","John",..: 2 3 6 4 1 5
 $ Gender   : Factor w/ 2 levels "F","M": 2 1 2 2 1 1
 $ Birthday : Factor w/ 6 levels "1968/10/15","1973/1/3",..: 2 6 4 1 5 3
 $ Income   : num  162.2 90.8 68.5 220.1 150 ...
 $ EventTime: Factor w/ 6 levels "02:30","05:20",..: 4 6 1 2 5 3
```

delimiter (定界符/分隔符號)

讀取具有日期及時間的資料

```
1;73;2017/01/27 11:30:20
2;52;2017/03/05 12:01:40
3;57;2017/05/12 03:20:00
1;74;2017/08/27 14:00:00
2;51;2017/10/17 21:03:50
3;60;2017/12/08 08:40:30
```

```
> myDT <- read.table("mydate.txt",
                     sep = ";")
> myDT
  V1 V2                  V3
1  1 73 2017/01/27 11:30:20
2  2 52 2017/03/05 12:01:40
3  3 57 2017/05/12 03:20:00
4  1 74 2017/08/27 14:00:00
5  2 51 2017/10/17 21:03:50
6  3 60 2017/12/08 08:40:30
> lapply(myDT, class)
$V1
[1] "integer"

$V2
[1] "integer"

$V3
[1] "factor"
```

```
> # 方法一
> varNames <- c("ID", "Values", "DateTime")
> myDT <- read.table("mydate.txt", sep = ";",
                     col.names = varNames)
> myDT
  ID Values             DateTime
1  1     73 2017/01/27 11:30:20
2  2     52 2017/03/05 12:01:40
3  3     57 2017/05/12 03:20:00
4  1     74 2017/08/27 14:00:00
5  2     51 2017/10/17 21:03:50
6  3     60 2017/12/08 08:40:30
> lapply(myDT, class)
$ID
[1] "integer"
$Values
[1] "integer"
$DateTime
[1] "factor"

> myDT$DateTime <- strptime(myDT$DateTime,
                            "%Y/%m/%d %H:%M:%S")
> lapply(myDT, class)
$ID
[1] "integer"
$Values
[1] "integer"
$DateTime
[1] "POSIXlt" "POSIXt"
```

設定自定的日期時間格式類別

```
> setClass('myDateTime') # 自定日期時間格式名稱
> setAs("character", "myDateTime",
        function(from) as.POSIXct(from, format="%Y/%m/%d %H:%M:%S"))
> varNames <- c("ID", "Values", "DateTime")
> varClasses <- c("integer", "numeric", "myDateTime")
> myDT <- read.table("mydate.txt", sep = ";", colClasses = varClasses,
                     col.names = varNames)
> myDT
  ID Values             DateTime
1  1     73 2017-01-27 11:30:20
2  2     52 2017-03-05 12:01:40
3  3     57 2017-05-12 03:20:00
4  1     74 2017-08-27 14:00:00
5  2     51 2017-10-17 21:03:50
6  3     60 2017-12-08 08:40:30
> lapply(myDT, class)
$ID
[1] "integer"

$Values
[1] "numeric"

$DateTime
[1] "POSIXct" "POSIXt"
```

```
1;73;2017/01/27 11:30:20
2;52;2017/03/05 12:01:40
3;57;2017/05/12 03:20:00
1;74;2017/08/27 14:00:00
2;51;2017/10/17 21:03:50
3;60;2017/12/08 08:40:30
```

注意事項

```
> read.table("input_testl.txt")
Error in file(file, "rt") : cannot open the connection
In addition: Warning message:
In file(file, "rt") :
  cannot open file 'input_testl.txt': No such file or directory
> read.table("input_test1.txt")
Error in scan(file = file, what = what, sep = sep, quote = quote, dec = dec,  :
  line 4 did not have 6 elements
> read.table("input_test1.txt", sep = "\t")
       V1 V2 V3 V4 V5 V6
1 subject x1 x2 x3 x4 x5
2      s1  a 90  1  F 11
3      s2  a 30  2  T 22
4      s3  b 20  5  T
5      s4  b 40  6  F 66
6      s5  c 20  7  T 77
>
> read.table("input_test1.txt", sep = "\t", header = T)
  subject x1 x2 x3    x4 x5
1      s1  a 90  1 FALSE 11
2      s2  a 30  2  TRUE 22
3      s3  b 20  5  TRUE NA
4      s4  b 40  6 FALSE 66
5      s5  c 20  7  TRUE 77
```

subject	x1	x2	x3	x4	x5
s1	a	90	1	F	11
s2	a	30	2	T	22
s3	b	20	5	T	
s4	b	40	6	F	66
s5	c	20	7	T	77

- Missing values:
  - code `"NA"` in the files.
  - `na.strings = "any words".`
  - Numeric columns: `NaN, Inf, -Inf`
- Blank lines:
  - `read.table()` ignores empty lines.
- Fixed-width-format file
  - `read.fwf()`
  - `read.fortran()`

讀取外部資料檔: scan( )

Description

Read data into a vector or list from the console or file.

Usage

```
scan(file = "", what = double(), nmax = -1, n = -1, sep = "",
     quote = if(identical(sep, "\n")) "" else "'\"", dec = ".",
     skip = 0, nlines = 0, na.strings = "NA",
     flush = FALSE, fill = FALSE, strip.white = FALSE,
     quiet = FALSE, blank.lines.skip = TRUE, multi.line = TRUE,
     comment.char = "", allowEscapes = FALSE,
     fileEncoding = "", encoding = "unknown", text, skipNul = FALSE)
```

sep

by default, scan expects to read white-space delimited input fields. Alternatively, sep can be used to specify a character which delimits fields. A field is always delimited by an end-of-line marker unless it is quoted.

skip

the number of lines of the input file to skip before beginning to read data values.

nlines

if positive, the maximum number of lines of data to be read.

讀取外部資料檔: scan()

```
my.data <- scan(file = "iris-data0.txt", what = list(w = numeric(0), x =
numeric(0), y = numeric(0), z = numeric(0), name = "character"))

my.mat <- as.data.frame(my.data)
```

```
my.data <- scan(file = "iris-data1.txt", what = list( n =integer(0), w =
numeric(0), x = numeric(0), y = numeric(0), z = numeric(0), name =
"character"), skip = 1)
my.data$n
```

iris-data0.txt

5.1	3.5	1.4	0.2	setosa
4.9	3	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa
5.4	3.7	1.5	0.2	setosa
4.8	3.4	1.6	0.2	setosa
4.8	3	1.4	0.1	setosa
4.3	3	1.1	0.1	setosa
5.8	4	1.2	0.2	setosa
5.7	4.4	1.5	0.4	setosa
5.4	3.9	1.3	0.4	setosa
5.1	3.5	1.4	0.3	setosa
5.7	3.8	1.7	0.3	setosa
5.1	3.8	1.5	0.3	setosa
5.4	3.4	1.7	0.2	setosa
5.1	3.7	1.5	0.4	setosa
4.6	3.6	1	0.2	setosa

iris-data1.txt

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa
11	5.4	3.7	1.5	0.2	setosa
12	4.8	3.4	1.6	0.2	setosa
13	4.8	3	1.4	0.1	setosa
14	4.3	3	1.1	0.1	setosa
15	5.8	4	1.2	0.2	setosa
16	5.7	4.4	1.5	0.4	setosa
17	5.4	3.9	1.3	0.4	setosa
18	5.1	3.5	1.4	0.3	setosa
19	5.7	3.8	1.7	0.3	setosa
20	5.1	3.8	1.5	0.3	setosa
21	5.4	3.4	1.7	0.2	setosa
22	5.1	3.7	1.5	0.4	setosa
23	4.6	3.6	1	0.2	setosa

課堂練習

Read in a large matrix

```
> getwd()
[1] "C:/Documents and Settings/user/My Documents"
> cat("1 2 3", "11 12 13", "21 22 23", "31 32 33", "41 42 43",
+ file = "ex.txt", sep = "\n")
> scan(file="ex.txt", what=list(x=0, y="", z=0))
Read 5 records
$x
[1]  1 11 21 31 41

$y
[1] "2"  "12" "22" "32" "42"

$z
[1]  3 13 23 33 43
```

```
1 2 3
11 12 13
21 22 23
31 32 33
41 42 43
```

Read in a large matrix

```
A <- matrix(scan("matrix.txt", n = 200 * 2000), 200, 2000, byrow = TRUE)
```

readLines()

```
readLines(con = stdin(), n = -1, ok = TRUE, warn = TRUE,
          encoding = "unknown")
```

Reading Large Data Files

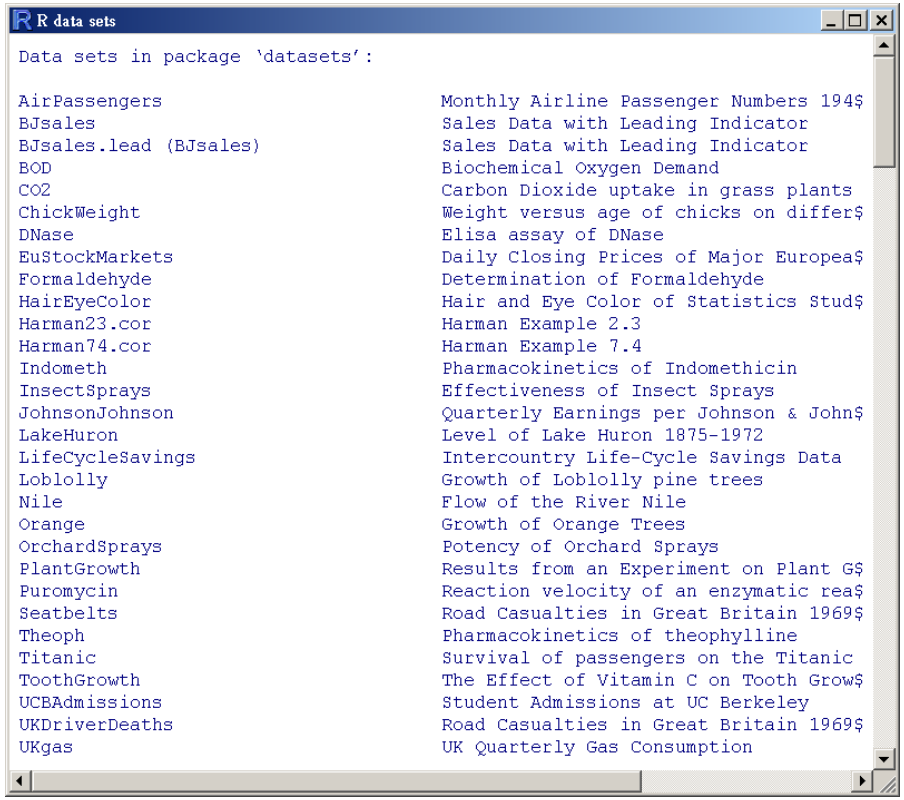
Since `readLines` and `scan` don't need to read an entire file into memory, there are situations where very large files can be processed by R in pieces.

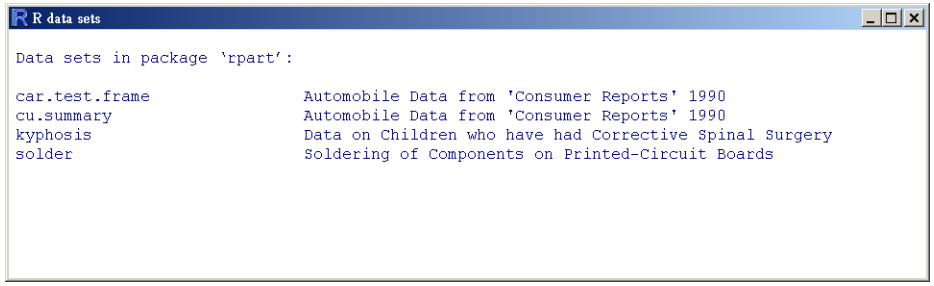
匯入R內建資料 (Load Builtin Data)

```
> data()

> data(Puromycin, package = “datasets”)
> Puromycin

> data(package = “rpart”)
```

讀取R的rda檔案：

```
> load(“test.rda")
```

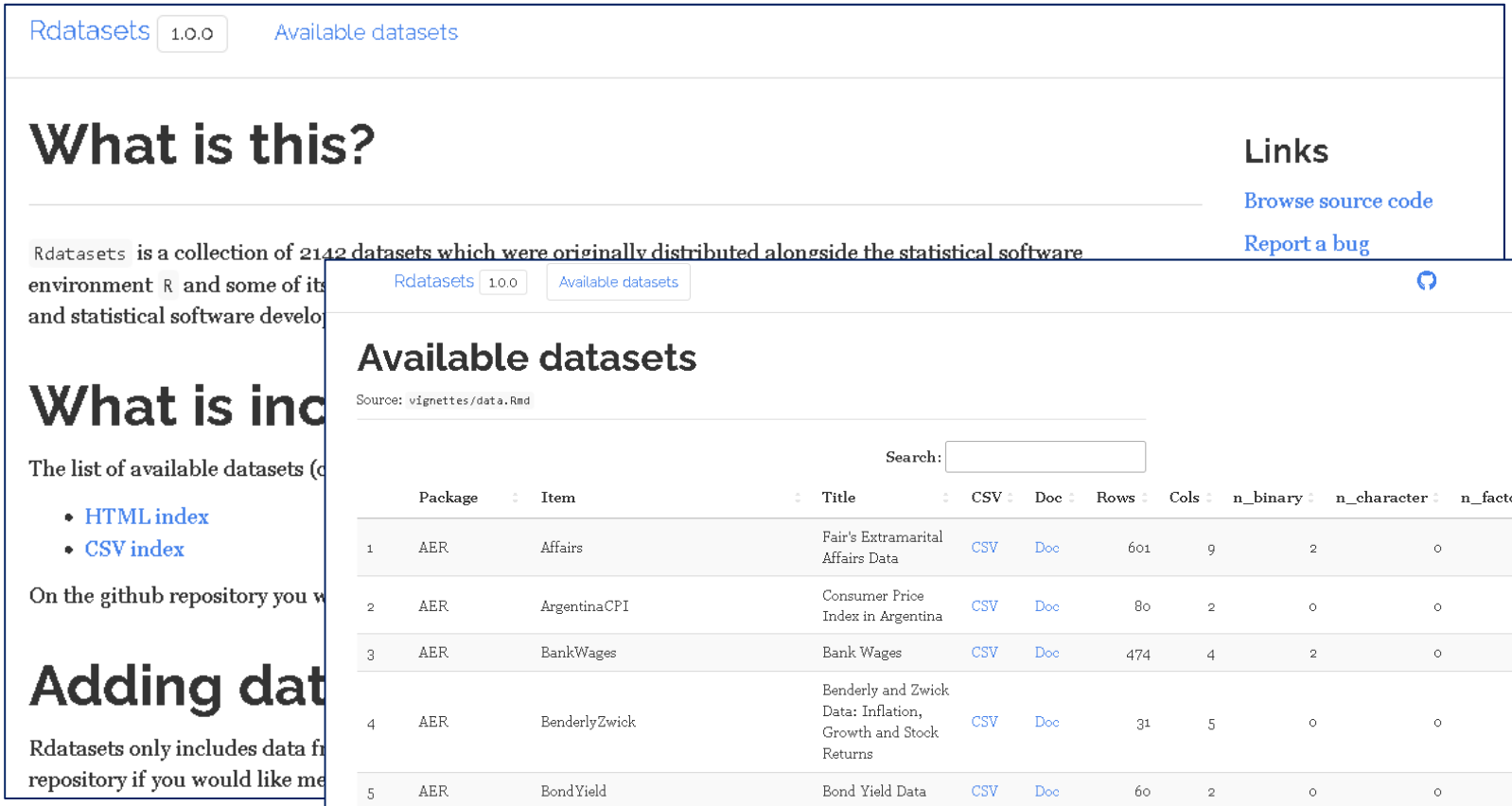





```
> library(MASS)
> data(crabs)
> ?crabs
> class(crabs)
> dim(crabs)
> colnames(crabs)
> str(crabs)
```

Rdatasets

https://vincentarelbundock.github.io/Rdatasets



Rdatasets 1.0.0 Available datasets

Available datasets

Source: `vignettes/data.Rmd`

Search:

	Package	Item	Title	CSV	Doc	Rows	Cols	n_binary	n_character	n_factor	n_logical	n_numeric
1	AER	Affairs	Fair's Extramarital Affairs Data	CSV	Doc	601	9	2	0	2	0	
2	AER	ArgentinaCPI	Consumer Price Index in Argentina	CSV	Doc	80	2	0	0	0	0	
3	AER	BankWages	Bank Wages	CSV	Doc	474	4	2	0	3	0	
4	AER	BenderlyZwick	Benderly and Zwick Data: Inflation, Growth and Stock Returns	CSV	Doc	31	5	0	0	0	0	
5	AER	BondYield	Bond Yield Data	CSV	Doc	60	2	0	0	0	0	
6	AER	CartelStability	CartelStability	CSV	Doc	328	5	2	0	3	0	
7	AER	CASchools	California Test Score Data	CSV	Doc	420	14	1	2	2	0	
8	AER	ChinaIncome	Chinese Real National Income Data	CSV	Doc	37	5	0	0	0	0	

編輯資料 (Editing Data)

```
library(MASS)
class(crabs)
dim(crabs)
colnames(crabs)
str(crabs)

#edit(data.name)
> edit(crabs)

#new.data <- edit(data.name)
> crabs.new <- edit(crabs)
> fix(crabs.new)

# new.data <- edit(data.frame())
> new.data <- edit(matrix(0, ncol=2, nrow=3))
```

Data Editor

	sp	sex	index	FL	RW	CL	CW	BD
1	B	M	1	8.1	6.7	16.1	19	7
2	B	M	2	8.8	7.7	18.1	20.8	7.4
3	B	M	3	9.2	7.8	19	22.4	7.7
4	B	M	4	9.6	7.9	20.1	23.1	8.2
5	B	M	5	9.8	8	20.3	23	8.2
6	B	M	6	10.8	9	23	26.5	9.8
7	B	M	7	11.1	9.9	23.8	27.1	9.8
8	B	M	8	11.6	9.1	24.5	28.4	10.4
9	B	M	9	11.8	9.6	24.2	27.8	9.7
10	B	M	10	11.8	10.5	25.2	29.3	10.3
11	B	M	11	12.2	10.8	27.3	31.6	10.9
12	B	M	12	12.3	11	26.8	31.5	11.4
13	B	M	13	12.6	10	27.7	31.7	11.4
14	B	M	14	12.8	10.2	27.2	31.8	10.9
15	B	M	15	12.8	10.9	27.4	31.5	11
16	B	M	16	12.9	11	26.8	30.9	11.4
17	B	M	17	13.1	10.6	28.2	32.3	11
18	B	M	18	13.1	10.9	28.3	32.4	11.2
19	B	M	19	13.3	11.1	27.8	32.3	11.3
20	B	M	20	13.9	11.1	29.2	33.3	12.1
21	B	M	21	14.3	11.6	31.3	35.5	12.7
22	B	M	22	14.6	11.3	31.9	36.4	13.7
23	B	M	23	15	10.9	31.4	36.4	13.2

匯出成資料檔 (Export to Text Files)

```
write.table(x, file = "", append = FALSE, quote = TRUE, sep = " ",
            eol = "\n", na = "NA", dec = ".", row.names = TRUE,
            col.names = TRUE, qmethod = c("escape", "double"))
```

header line

```
> write.csv(iris, "myNewData.csv", sep = ",", col.names = TRUE)

> write.table(iris, "myNewData.txt", quote = FALSE, sep = "\t")
```

```
> library(MASS)
> hills
> hills10 <- hills[1:10, 1:2]
> hills10

> write.table(hills10, "hill10.txt", sep = "\t", quote = F, row.names = TRUE)

> write.table(hills[11:15,1:2], "hill10.txt", append = TRUE, sep = "\t", row.names =
TRUE, col.names = FALSE)
```

Note: 在既有的資料檔案中，加入資料時，需要有相同的欄位名稱。

sink {base}: Send R Output to a File

```
>  zz <- file("output.txt", "w")
>  cat("Title line", "2 3 5 7", " ", "11 13 17", file = zz, sep = "\n")
>  cat("One more line \n", file = zz)
>  close(zz)


>  zz <- textConnection("output.obj", "w")
>  sink(zz)
>  example(lm)
>  sink()
>  close(zz)
>  cat(output.obj, sep = "\n")
>  write(output.obj, file = "result.txt")
```

課堂練習

```
> iris[1:10, ]
> write.table(iris, "iris-data0.txt", sep = "\t", quote = F, row.names = FALSE, col.names = FALSE)
> write.table(iris, "iris-data1.txt", sep = "\t", quote = F, row.names = TRUE, col.names = TRUE)

> write.table(hills[11:15,1:2], "iris-data2.txt", append = TRUE, sep = "\t", row.names = TRUE, col.names = FALSE)

> write.table(hills[11:15,1:2], "iris-data3.txt", append = TRUE, sep = "\t", row.names = TRUE, col.names = FALSE)
```

iris-data0.txt

5.1	3.5	1.4	0.2	setosa
4.9	3	1.4	0.2	setosa
4.7	3.2			
4.6	3.1			
5	3.6			
5.4				
4.6				
5				
4.4				
4.9				
5.4				
4.8				
4.8				
4.3				
5.8				
5.7				
5.4				
5.1				
5.7				
5.1				
5.4				
5.1				
4.6				

iris-data1.txt

	Sep...				...pcies
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa
11	5.4	3.7	1.5	0.2	setosa
12	4.8	3.4	1.6	0.2	setosa
13	4.8	3	1.4	0.1	setosa
14	4.3	3	1.1	0.1	setosa
15	5.8	4	1.2	0.2	setosa
16	5.7	4.4	1.5	0.4	setosa
17	5.4	3.9	1.3	0.4	setosa
18	5.1	3.5	1.4	0.3	setosa
19	5.7	3.8	1.7	0.3	setosa
20	5.1	3.8	1.5	0.3	setosa
21	5.4	3.4	1.7	0.2	setosa
22	5.1	3.7	1.5	0.4	setosa
23	4.6	3.6	1	0.2	setosa

iris-data2.txt

no	Sepal.Length	Sepal.Width	Petal.Length	F
1	5.1	3.5	1.4	
2	4.9	3		
3	4.7	3.2		
4	4.6	3.1		
5	5	3.6		
6	5.4	3.9		
7	4.6	3.4		
8	5	3.4		
9	4.4	2.9		
10	4.9	3.1		
11	5.4	3.7		
12	4.8	3.4		
13	4.8	3		
14	4.3	3		
15	5.8	4		
16	5.7	4.4		
17	5.4	3.9		
18	5.1	3.5		
19	5.7	3.8		
20	5.1	3.8		
21	5.4	3.4		
22	5.1	3.7		
23	4.6	3.6		

iris-data3.txt

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa
5.4	3.7	1.5	0.2	setosa
4.8	3.4	1.6	0.2	setosa
4.8	3	1.4	0.1	setosa
4.3	3	1.1	0.1	setosa
5.8	4	1.2	0.2	setosa
5.7	4.4	1.5	0.4	setosa
5.4	3.9	1.3	0.4	setosa
5.1	3.5	1.4	0.3	setosa
5.7	3.8	1.7	0.3	setosa
5.1	3.8	1.5	0.3	setosa
5.4	3.4	1.7	0.2	setosa
5.1	3.7	1.5	0.4	setosa
4.6	3.6	1	0.2	setosa

課堂練習

```
> my.data0 <- read.table("iris-data0.txt")
> my.data0[1:5, ] # or head(mydata0)

> my.data1 <- read.table("iris-data1.txt")
> my.data1[1:5, ]

> my.data2 <- read.table("iris-data2.txt", header = TRUE, row.names = 1)
> my.data2[1:5, ]

> my.data3 <- read.table("iris-data3.txt", header = TRUE, sep = "\t")
> my.data3[1:5, ]
```

```
> my.sdata0 <- scan(file = "iris-data0.txt", what = list(w = numeric(0), x = numeric(0),
y = numeric(0), z = numeric(0), name = "character"))
> my.sdata0
> my.mat <- as.data.frame(my.data)
> my.mat[1:5, ]
```

```
> my.sdata1 <- scan(file = "iris-data1.txt", what = list(n = integer(0), w = numeric(0),
x = numeric(0), y = numeric(0), z = numeric(0), name = "character"), skip = 1)
> str(my.sdata1)
> my.sdata1$n
```

讀取部份資料

- 僅輸入所需要的部份資料，而不是全部。

```
Variables <- c("NULL", "NULL", "factor", "numeric")
myData <- read.table("fileName", colClasses = Variables)
```

- 用適合的函式或演算法: O(N) vs O(N^2)

```
x <- 1:10000; s <- sample(x, 10)
a1 <- which(x %in% s)
a2 <- intersect(x, s)
a3 <- which(is.element(x, s))

for(i in 1:10000){
    for(j in 1:10){
       if(all.equal(x[i], s[j]){
           ...
       }
    }
}
```

```
> n <- 10000
> p <- 1000
> Mat <- matrix(rnorm(n*p), nrow = n, ncol = p)
> system.time(apply(Mat, 1, sum))
   user  system elapsed
   0.61    0.19    2.56
> system.time(rowSums(Mat))
   user  system elapsed
   0.05    0.00    0.08
```

See also: CRAN Task View: High-Performance and Parallel Computing with R

二進位儲存資料

- 資料儲存以二進位檔(binary)為優先:
  - 讀寫文字檔比壓縮二進位檔慢。
  - 壓縮二進位檔又比二進位慢。

```
> n <- 1000
> p <- 1000
> Mat <- matrix(rnorm(n*p),
nrow = n, ncol = p)
```

```
> system.time(write.table(Mat, file = "myData.txt"))
   user  system elapsed
   8.89    0.09   12.14
> system.time(read.table("myData.txt"))
   user  system elapsed
  10.85    0.06   11.98
```

```
> system.time(save(Mat, file = "myData.gz"))
   user  system elapsed
   1.11    0.01    2.52
> system.time(load("myData.gz"))
   user  system elapsed
   0.36    0.02    3.56
```

```
> system.time(save(Mat, file = "myData.Rdata", compress = FALSE))
   user  system elapsed
   0.24    0.00    0.23
> system.time(load("myData.Rdata"))
   user  system elapsed
   0.23    0.00    0.24
```

讀取 XML 檔案:
xmlToDataFrame {XML}

```
> library(XML)
> sample.data <- xmlToDataFrame("Sample-XML-Files.xml")
> str(sample.data)
'data.frame':   3 obs. of  6 variables:
 $ TITLE  : chr  "dill diya galla" "Saiyara" "Khairiyat"
 $ ARTIST : chr  "Arijit singh" "Atif Aslam" "Sonu nigam"
 $ COUNTRY: chr  "India" "Uk" "india"
 $ COMPANY: chr  "tseries" "Records" "radio"
 $ PRICE  : chr  "10.90" "9.90" "9.90"
 $ YEAR   : chr  "2018" "2015" "2019"
> head(sample.data)
            TITLE       ARTIST COUNTRY COMPANY PRICE YEAR
1 dill diya galla Arijit singh   India tseries 10.90 2018
2         Saiyara   Atif Aslam      Uk Records  9.90 2015
3       Khairiyat   Sonu nigam   india   radio  9.90 2019
```

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type='text/xsl'?>
<CATALOG>
    <CD>
    <TITLE>dill diya galla</TITLE>
    <ARTIST>Arijit singh</ARTIST>
    <COUNTRY>India</COUNTRY>
    <COMPANY>tseries</COMPANY>
    <PRICE>10.90</PRICE>
    <YEAR>2018</YEAR>
    </CD>
    <CD>
    <TITLE>Saiyara</TITLE>
    <ARTIST>Atif Aslam</ARTIST>
    <COUNTRY>Uk</COUNTRY>
    <COMPANY>Records</COMPANY>
    <PRICE>9.90</PRICE>
    <YEAR>2015</YEAR>
    </CD>
    <CD>
    <TITLE>Khairiyat</TITLE>
    <ARTIST>Sonu nigam</ARTIST>
    <COUNTRY>india</COUNTRY>
    <COMPANY>radio</COMPANY>
    <PRICE>9.90</PRICE>
    <YEAR>2019</YEAR>
    </CD>
</CATALOG>
```

XML [編輯]

維基百科，自由的百科全書

可延伸標記式語言（英語：Extensible Markup Language，簡稱：**XML**）是一種標記式語言。標記指電腦所能理解的資訊符號，通過此種標記，電腦之間可以處理包含各種資訊的文章等。如何定義這些標記，既可以選擇國際通用的標記式語言，比如HTML，也可以使用像XML這樣由相關人士自由決定的標記式語言，這就是語言的可延伸性。XML是從標準通用標記式語言（SGML）中簡化修改出來的。它主要用到的有可延伸標記式語言、可延伸樣式語言（XBRL和XPath等。

例 [編輯]

XML定義結構、儲存資訊、傳送資訊。下例為小張傳送給大元的便條，儲存為XML。

```
<?xml version="1.0"?>
<小紙条>
  <收件人>大元</收件人>
  <發件人>小張</發件人>
  <主題>問候</主題>
  <具體內容>早啊，飯吃了沒？ </具體內容>
</小紙条>
```

這XML文件僅是純粹的資訊標籤，這些標籤意義的展開依賴於應用它的程式。

- 維基百科: XML: https://zh.wikipedia.org/zh-tw/XML
- XML Note: https://irw.ncut.edu.tw/peterju/xml.html
- Sample file: https://www.learningcontainer.com/sample-xml-file/

讀取JSON資料檔:
fromJSON {jsonlite}

JSON [編輯]

維基百科，自由的百科全書

JSON（JavaScript Object Notation，JavaScript物件表示法，讀作/ˈdʒeɪsən/）是一種由道格拉斯·克羅克福特構想和設計、輕量級的資料交換語言，該語言以易於讓人閱讀的文字為基礎，用來傳輸由屬性值或者序列性的值組成的資料物件。儘管JSON是JavaScript的一個子集，但JSON是獨立於語言的文字格式，並且採用了類似於C語言家族的一些習慣。

JSON 資料格式與語言無關。即便它源自JavaScript，但目前很多程式語言都支援 JSON 格式資料的生成和解析。JSON 的官方 MIME 類型是 `application/json`，副檔名是 `.json`。

```
> library(jsonlite)
> my.df <- fromJSON("Hsinchu_Death_Top10_108.json")
> head(my.df)
  順位                 全部死亡原因 全部死亡率-每十萬人口
1    1                       惡性腫瘤              162.9
2    2 心臟疾病（高血壓性疾病除外）               72.3
3    3                     腦血管疾病               59.1
4    4                       糖尿病               47.8
5    5                         肺炎               47.6
6    6                   高血壓性疾病               34.8
```

Hsinchu_Death_Top10_108.json


```
[
 {
  "順位": "1",
  "全部死亡原因": "惡性腫瘤",
  "全部死亡率-每十萬人口": "162.9",
  "男性死亡原因": "惡性腫瘤",
  "男性死亡率-每十萬男性人口": "189.4",
  "女性死亡原因": "惡性腫瘤",
  "女性死亡率-每十萬女性人口": "135.2"
 },

...

 {
  "順位": "10",
  "全部死亡原因": "慢性肝病及肝硬化",
  "全部死亡率-每十萬人口": "17",
  "男性死亡原因": "慢性肝病及肝硬化",
  "男性死亡率-每十萬男性人口": "23.8",
  "女性死亡原因": "腎炎、腎病症候群及腎病變",
  "女性死亡率-每十萬女性人口": "13.5"
 }
]
```


- fromJSON 將基本資料類型(字串、數值、布林值或 null)的 JSON 陣列，轉換為 R 的向量。
- 具有多個物件的 JSON 資料，fromJSON 會將其轉換為 R 的 data frame。
- 包含二維陣列的 JSON 資料時，fromJSON會轉換為 R 的矩陣。
- 高維度的 JSON 陣列，fromJSON會轉換為 R 的陣列。

讀取其它軟體資料檔案: foreign Package

- This is often best avoided!

```
> read.xport()  # SAS XPORT
> read.ssd() # SAS dataset
> read.S() # S-plus binary object
> read.spss() # SPSS
> read.xls() # R package(xlsReadWrite)
```

匯入SPSS (.sav)(read.spss函式不支援中文，如果遇到.sav檔中有中文則必須要從SPSS中匯出成CSV後再從R把CSV匯入)。

Function(s)	Purpose
`data.restore` `read.S`	read `data.dump` output or saved objects from S version 3 may work with older Splus objects
`read.dbf`	read or write saved objects from DBF files (FoxPro, dBase, etc.)
`read.dta` `write.dta`	read saved objects from Stata (versions 5-9) create a Stata saved object
`read.epinfo`	read saved objects from epinfo
`read.spss`	read saved objects from SPSS written using the `save` or `export` command
`read.mtp`	read Minitab Portable Worksheet files
`read.octave`	read saved objects from GNU octave
`read.xport`	read saved objects in SAS export format
`read.systat`	read saved objects from systat rectangular (mtype=1) data only

Table 2.3. Functions in the `foreign` package

- Browsing to find files

```
> Data <- read.table(file.choose(), header=TRUE)
```

- Checking files from the command line

```
> File.exists("c:\\temp\\data.txt")
```

讀取SPSS檔案 (*.sav): read.spss {foreign}

```
> library(foreign)
> dataset <- read.spss("electric.sav", to.data.frame=TRUE)
> dim(dataset)
[1] 240  13
> head(dataset)
  CASEID       FIRSTCHD AGE DBP58 EDU
1     13     NONFATALMI  40    70
...
6    102     NONFATALMI  50    88
```

*electric.sav [DataSet1] — PSPPIRE Data Editor

File Edit View Data Transform Analyze Graphs Utilities Windows Help

Case	caseid	firstchd	age	dbp58	eduyr	chol58	cgt58	ht58	wt58	dayofwk	vital10	famhxcvr	chd
1	13	3	40	70	16	321	0	68.8	190	9	0	Y	1
2	30	3	49	87	11	246	60	72.2	204	5	0	N	1
3	53	2	43	89	12	262	0	69.0	162	7	1	N	1
4	84	3	50	105	8	275	15	62.5	152	4	0	Y	1
5	89	2	43	110	.	301	25	68.0	148	2	1	N	1
6	102	3	50	88	8	261	30	68.0	142	1	1	N	1
7	117	3	45	70	.	212	0	66.5	196	9	0	N	1

```
> dataset2 <- read.spss("test-spss-data.sav", to.data.frame=TRUE)
Error in read.spss("test-spss-data.sav", to.data.frame = TRUE) :
  error reading system-file header
此外: Warning message:
In read.spss("test-spss-data.sav", to.data.frame = TRUE) :
  test-spss-data.sav: position 0: Variable name begins with invalid character
```

*test-spss-data.sav [DataSet3] — PSPPIRE Data Editor

File Edit View Data Transform Analyze Graphs Utilities Windows Help

Case	−.1	−.2	−.3	服務品質	品牌形象
1	1	8	2	20.00	16.00
2	2	7	3	28.00	21.00
3	1	7	3	27.00	18.00
4	2	2	3	29.00	21.00
5	1	9	4	25.00	16.00
6	2	8	2	27.00	18.00
7	2	3	4	29.00	18.00

Data View Variable View

Filter off Weights off No Split

GNU PSPP is a program for statistical analysis of sampled data. It is a free as in freedom replacement for the proprietary program SPSS, and appears very similar to it with a few exceptions.
https://www.gnu.org/software/pspp/

讀取SPSS檔案 (*.sav):

spss.system.file {memisc}

```
> library(memisc)
> dataset2 <- as.data.set(spss.system.file("test-spss-data.sav"))
> dim(dataset2)
[1] 12  5
> head(dataset2)
Data set with 12 observations and 5 variables
   一.1     一.2 一.3 服務品質 品牌形象
 1    1  56-60歲    2       20       16
 2    2  51-55歲    3       28       21
 3    1  51-55歲    3       27       18
 ...
12    2  26-30歲    4       22       16
> str(dataset2)
Data set with 12 obs. of 5 variables:
 $ 一.1    : Nmnl. item w/ 2 labels for 6.01347001699909e-154,6.01347001699909e-154  chr
"1" "2" "1" "2" ...
..
 $ 服務品質: Itvl. item  num  20 28 27 29 25 27 29 27 27 20 ...
 $ 品牌形象: Itvl. item  num  16 21 18 21 16 18 18 18 24 16 ...
> dataset2$一.1 #第一個欄位資料
Item '性別' (measurement: nominal, type: character, length = 12)
 [1:12] 1 2 1 2 1 2 2 1 2 2 1 2
> dataset2$服務品質
Item (measurement: interval, type: double, length = 12)
 [1:12] 20 28 27 29 25 27 29 27 27 20 29 22
```

See also: `read_sav {haven}`, `read_spss{haven}`

讀取Excel資料檔案

read.xlsx {xlsx}

```
read.xlsx(file, sheetIndex, sheetName=NULL, rowIndex=NULL,
          startRow=NULL, endRow=NULL, colIndex=NULL,
          as.data.frame=TRUE, header=TRUE, colClasses=NA,
          keepFormulas=FALSE, encoding="unknown", ...)
```

不建議使用**xlsx**套件。
請使用**readxl**套件。

- **rowIndex (colIndex)**: a numeric vector indicating the rows (cols) you want to extract.
- **header**: a logical value indicating whether the first row corresponding to the first element of the rowIndex vector contains the names of the variables.
- **colClasses**: a character vector that represent the class of each column. (numeric, character, Date, POSIXct)
- **keepFormulas**: a logical value indicating if Excel formulas should be shown as text in R and not evaluated before bringing them in.
- **encoding**: encoding to be assumed for input strings.

若**library(xlsx)**時，load rJava 有問題，解決方式如下:
首先，確R和Java(jdk-8u101-windows-x64.exe)都是64位元的。

```
> version
> packageVersion('rJava')
```

在R中設定Java的路徑。

```
> Sys.getenv("JAVA_HOME")
> Sys.setenv(JAVA_HOME='C:\\Program Files\\
Java\\jdk1.8.0_45\\jre')
```

重新安裝xlsx和rJava套件。

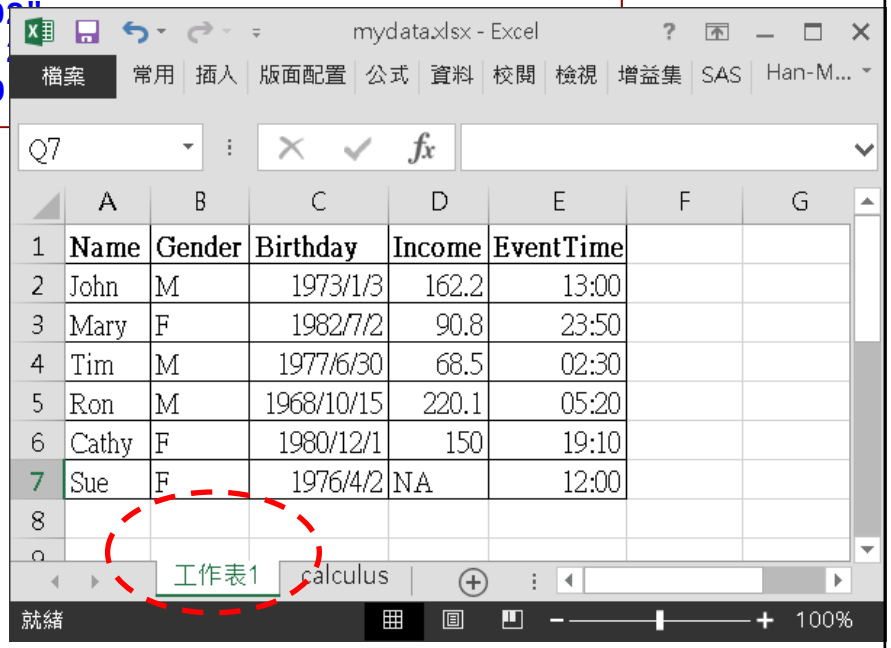
```
> install.packages("xlsx")
> install.packages("rJava")
```

重新啟動R，並載入xlsx套件即可。

```
> library(xlsx)
```

讀取Excel資料檔案 (xlsx套件)

```
> library(xlsx)
> mydata.sheet1 <- read.xlsx("mydata.xlsx", 1)
> head(mydata.sheet1)
   Name Gender   Birthday Income           EventTime
1  John      M 1973-01-03  162.2 1899-12-30 13:00:00
2  Mary      F 1982-07-02   90.8 1899-12-30 23:50:00
3   Tim      M 1977-06-30   68.5 1899-12-30 02:30:00
4   Ron      M 1968-10-15  220.1 1899-12-30 05:20:00
5 Cathy      F 1980-12-01    150 1899-12-30 19:10:00
6   Sue      F 1976-04-02     NA 1899-12-30 12:00:00
> str(mydata.sheet1)
'data.frame':   6 obs. of  5 variables:
 $ Name     : Factor w/ 6 levels "Cathy","John",..: 2 3 6 4 1 5
 $ Gender   : Factor w/ 2 levels "F","M": 2 1 2 2 1 1
 $ Birthday : Date, format: "1973-01-03" "1982-07-02"
 $ Income   : Factor w/ 6 levels "150","162.2",..: 
 $ EventTime: POSIXct, format: "1899-12-30 13:00:00
```

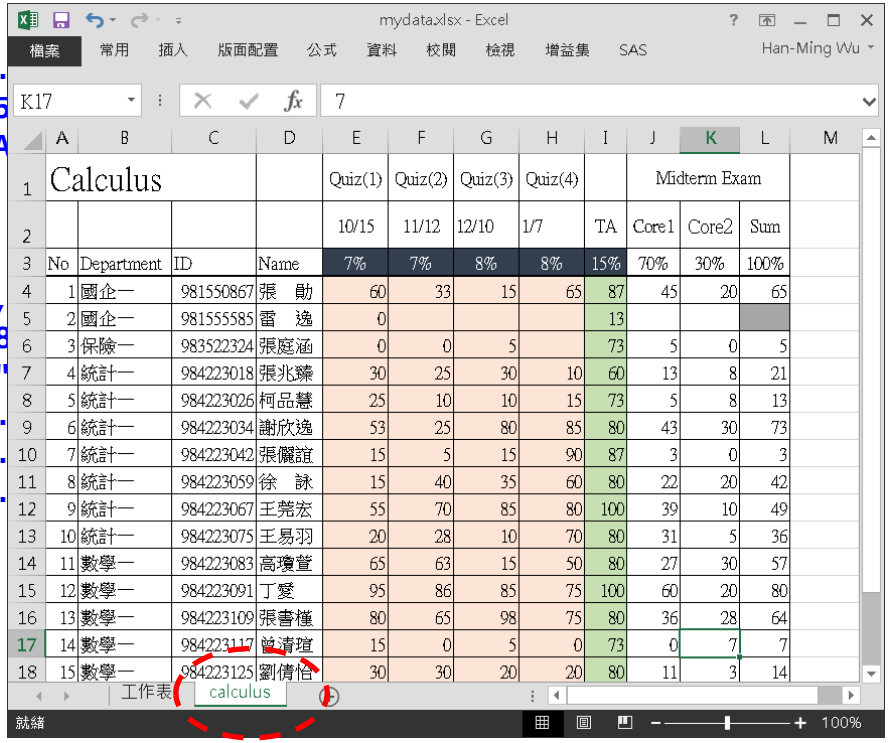


See also:

```
library(XLConnect)
df <- readWorksheetFromFile("<file name and
extension>", sheet = 1)
```

讀取/寫出Excel資料檔案 (xlsx套件)

```
> myCol <- c("integer", NA, rep("character", 2), rep("numeric", 8))
> mydata.sheet2 <- read.xlsx("mydata.xlsx", 2, startRow=3,
+                            header=TRUE, encoding="UTF-8",
+                            colClasses=myCol)
> head(mydata.sheet2, 2)
  No Department        ID   Name X0.07 X0.07.1 X0.
1  1     國企一 981550867 張　勛    60      33    15
2  2     國企一 981555585 雷　逸     0      NA    NA
> str(mydata.sheet2)
'data.frame':   19 obs. of  12 variables:
 $ No        : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Department: Factor w/ 4 levels "保險一","國企一",
 $ ID        : Factor w/ 19 levels "981550867","98
 $ Name      : Factor w/ 19 levels "丁愛 ","王易羽"
 $ X0.07     : num  60 0 0 30 25 53 15 15 55 20 ..
 $ X0.07.1   : num  33 NA 0 25 10 25 5 40 70 28 ..
 $ X0.08     : num  15 NA 5 30 10 80 15 35 85 10 .
...
```



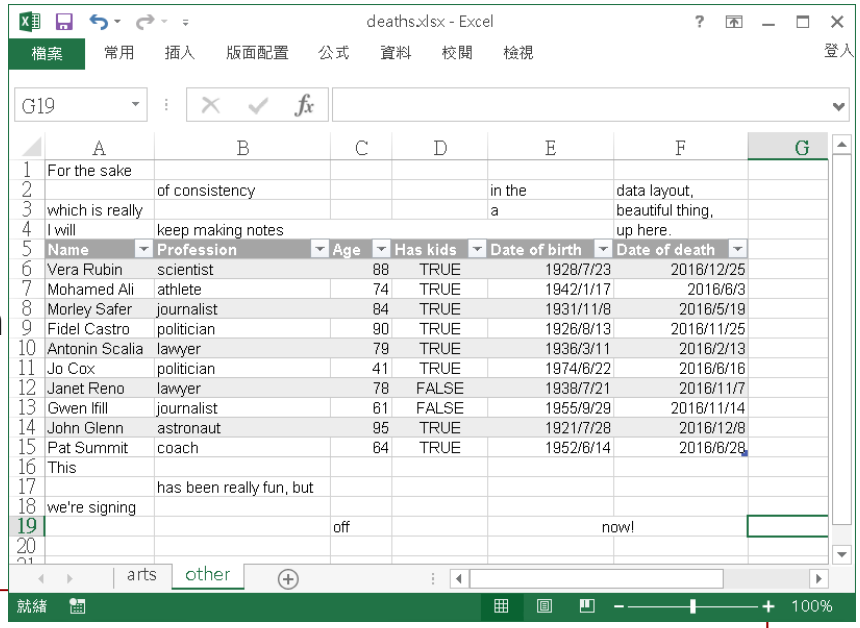

```
> colnames(mydata.sheet2) <- c(colnames(mydata.sheet2)[1:4],
paste("Quiz", 1:4, sep=""), "TA", "MidCore1", "MidCore2", "MidSum")
> head(mydata.sheet2, 2)
  No Department        ID   Name Quiz1 Quiz2 Quiz3 Quiz4 TA MidCore1 MidCore2 MidSum
1  1     國企一 981550867 張　勛    60    33    15    65 87       45       20      65
2  2     國企一 981555585 雷　逸     0    NA    NA    NA 13       NA       NA      NA
> write.xlsx(mydata.sheet2, "calculus.xlsx")
```

讀取Excel資料檔案

read_excel {readxl}

Features of **readxl:**

- No external dependency (e.g., Java or Perl).
- Re-encodes non-ASCII characters to UTF-8.
- Loads datetimes into POSIXct columns.
- More control with **range**, **skip**, and **n_max**.
- Column names and types are determined from the data in the sheet, by default.
- User can also supply via **col_names** and **col_types**.

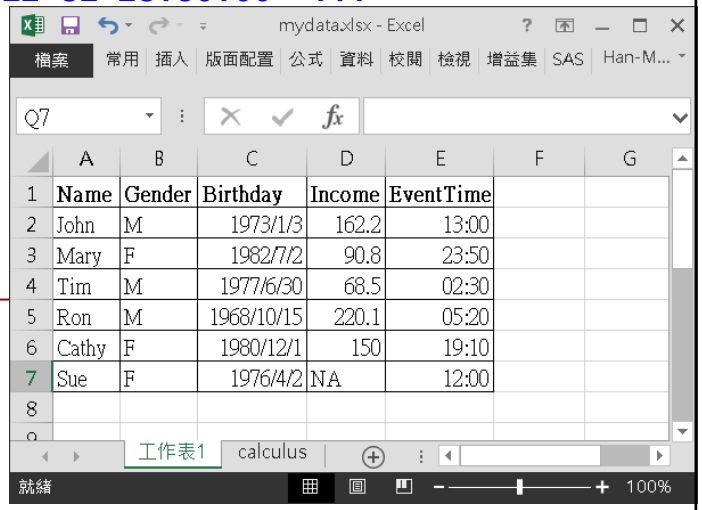


	A	B	C	D	E	F	G
1	For the sake						
2		of consistency			in the	data layout,	
3	which is really				a	beautiful thing,	
4	I will	keep making notes				up here.	
5	Name	Profession	Age	Has kids	Date of birth	Date of death	
6	Vera Rubin	scientist	88	TRUE	1928/7/23	2016/12/25	
7	Mohamed Ali	athlete	74	TRUE	1942/1/17	2016/6/3	
8	Morley Safer	journalist	84	TRUE	1931/11/8	2016/5/19	
9	Fidel Castro	politician	90	TRUE	1926/8/13	2016/11/25	
10	Antonin Scalia	lawyer	79	TRUE	1936/3/11	2016/2/13	
11	Jo Cox	politician	41	TRUE	1974/6/22	2016/6/16	
12	Janet Reno	lawyer	78	FALSE	1938/7/21	2016/11/7	
13	Gwen Ifill	journalist	61	FALSE	1955/9/29	2016/11/14	
14	John Glenn	astronaut	95	TRUE	1921/7/28	2016/12/8	
15	Pat Summit	coach	64	TRUE	1952/6/14	2016/6/28	
16	This						
17		has been really fun, but					
18	we're signing						
19			off			now!	
20							

```
> library(readxl)
> readxl_example()
 [1] "clippy.xls"    "clippy.xlsx"   "datasets.xls"  "datasets.xlsx" "deaths.xls"
 [6] "deaths.xlsx"   "geometry.xls"  "geometry.xlsx" "type-me.xls"   "type-me.xlsx"
> xlsx_example <- readxl_example("datasets.xlsx")
> xlsx_example
[1] "C:/Users/userpc/Documents/R/win-library/3.4/readxl/extdata/datasets.xlsx"
> mydata <- read_excel(xlsx_example) # reads both xls and xlsx.
> head(mydata, 3)
# A tibble: 6 x 3
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
         <dbl>       <dbl>        <dbl>       <dbl>   <chr>
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
```

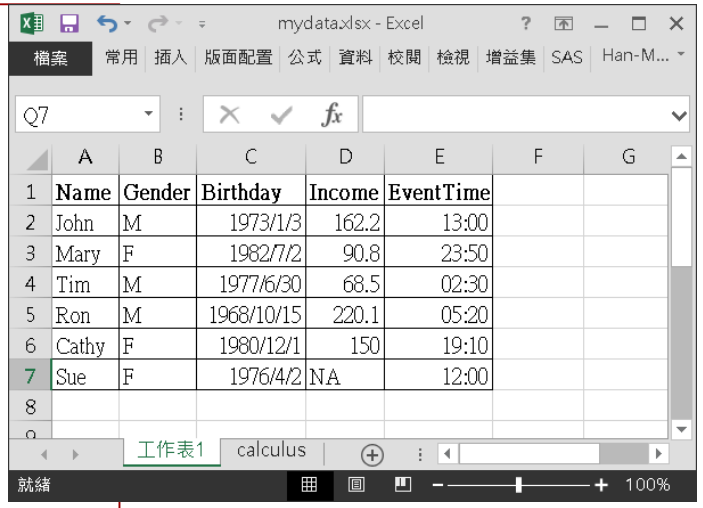
read_excel: More Controls

```
> xlsx_file <- "mydata.xlsx"
> excel_sheets(xlsx_file) # List the sheet names
[1] "工作表1"  "calculus"
> mydata <- read_excel(xlsx_file, sheet = "工作表1", na = "NA")
> head(mydata, 3)
# A tibble: 3 x 5
   Name Gender   Birthday Income           EventTime
  <chr>  <chr>     <dttm>  <dbl>              <dttm>
1  John      M 1973-01-03  162.2 1899-12-31 13:00:00
2  Mary      F 1982-07-02   90.8 1899-12-31 23:50:00
3   Tim      M 1977-06-30   68.5 1899-12-31 02:30:00
> str(mydata)
Classes ‘tbl_df’, ‘tbl’ and 'data.frame':        6 obs. of  5 variables:
 $ Name     : chr  "John" "Mary" "Tim" "Ron" ...
 $ Gender   : chr  "M" "F" "M" "M" ...
 $ Birthday : POSIXct, format: "1973-01-03" "1982-07-02" ...
 $ Income   : num  162.2 90.8 68.5 220.1 150 ...
 $ EventTime: POSIXct, format: "1899-12-31 13:00:00" "1899-12-31 23:50:00" ...
> read_excel(xlsx_file, n_max = 3, na = "NA")
# A tibble: 3 x 5
   Name Gender   Birthday Income           EventTime
  <chr>  <chr>     <dttm>  <dbl>              <dttm>
1  John      M 1973-01-03  162.2 1899-12-31 13:00:00
2  Mary      F 1982-07-02   90.8 1899-12-31 23:50:00
3   Tim      M 1977-06-30   68.5 1899-12-31 02:30:00
```



read_excel: More Controls

```
> read_excel(xlsx_file, range = "C1:E4")
# A tibble: 3 x 3
    Birthday Income           EventTime
      <dttm>  <dbl>              <dttm>
1 1973-01-03  162.2 1899-12-31 13:00:00
2 1982-07-02   90.8 1899-12-31 23:50:00
3 1977-06-30   68.5 1899-12-31 02:30:00
> read_excel(xlsx_file, range = cell_rows(1:4))
# A tibble: 3 x 5
   Name Gender   Birthday Income           EventTime
  <chr>  <chr>     <dttm>  <dbl>              <dttm>
1  John      M 1973-01-03  162.2 1899-12-31 13:00:00
2  Mary      F 1982-07-02   90.8 1899-12-31 23:50:00
3   Tim      M 1977-06-30   68.5 1899-12-31 02:30:00
> read_excel(xlsx_file, range = cell_cols("B:D"), na = "NA")
# A tibble: 6 x 3
  Gender   Birthday Income
   <chr>     <dttm>  <dbl>
1      M 1973-01-03  162.2
2      F 1982-07-02   90.8
3      M 1977-06-30   68.5
4      M 1968-10-15  220.1
5      F 1980-12-01  150.0
6      F 1976-04-02     NA
```




```
skip = 5
col_types = c("date", "skip", "guess", "numeric",
              "text", "list", "logical")
```

See also:

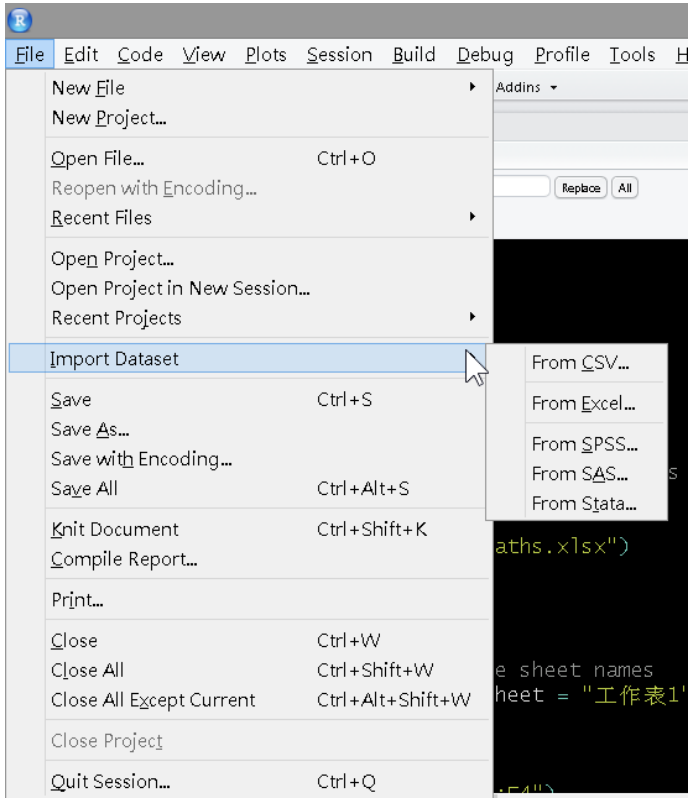
http://readxl.tidyverse.org/articles/articles/readxl-workflows.html

http://readxl.tidyverse.org/articles/sheet-geometry.html

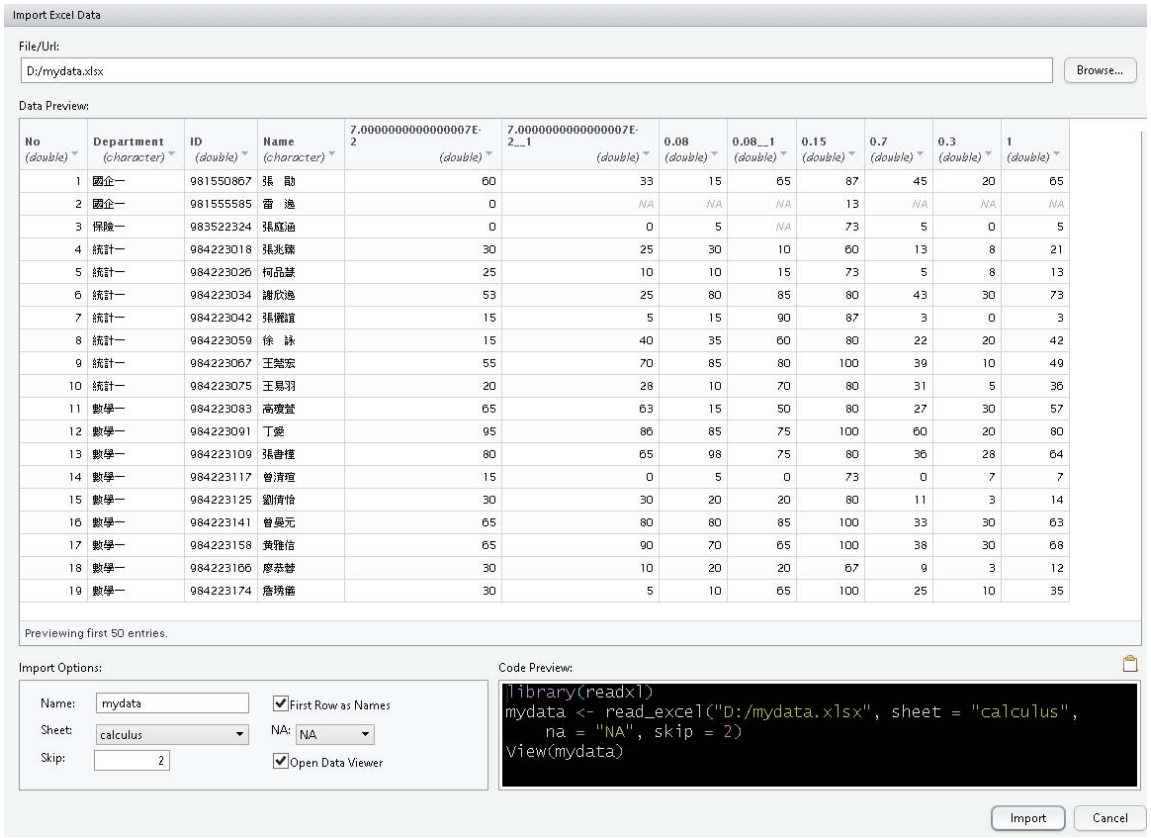
```
# write data to a excel file
> outdata <- list(iris = iris, airquality = airquality)
> library(openxlsx)
> write.xlsx(outdata, file = "outdata.xlsx")
```

write_xlsx {writexl}: Export Data Frames to Excel 'xlsx' Format

利用RStudio匯入資料: CSV、Excel、SPSS、SAS、Stata



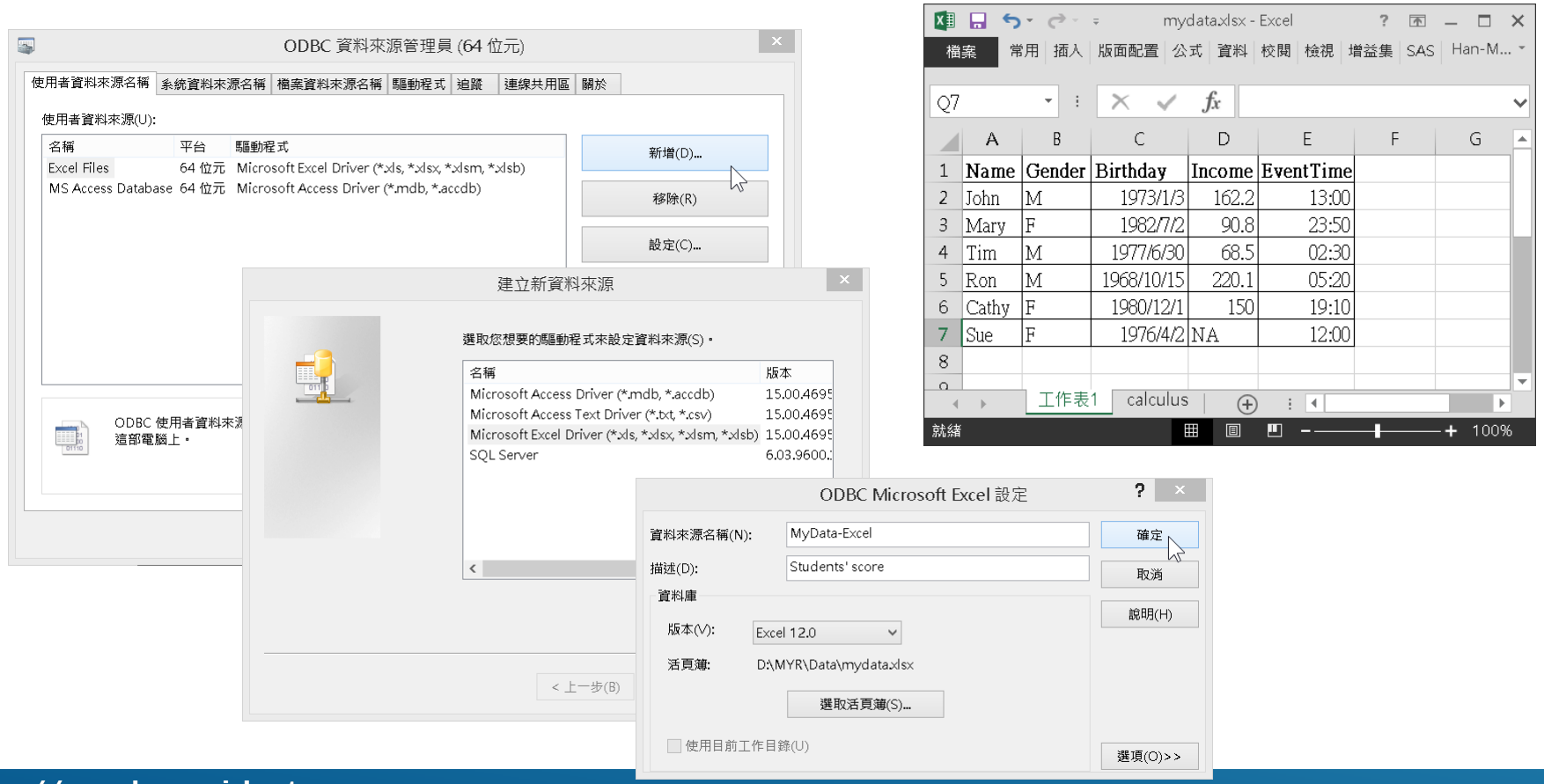

```
> library(haven)
> math <- read_sav("D:/math.sav") # read spss data file
> View(math)
> meat <- read_sas("D:/meat.sas7bdat")
> View(meat)
```



NOTE: 不要用中文目錄名。

在R中使用ODBC 讀取 Excel 檔案 (Windows為例)

- Open Data Base Connectivity (ODBC) is a protocol that allows access to database systems (and spreadsheets) that implement it. The protocol is common and is implemented in package RODBC.
- STEP(1): Name a connection: 控制台 => 系統管理工具 => ODBC 資料來源(64位元) => ODBC 資料來源管理員(64位元) => 新增 => 建立新資料來源 => 選「Microsoft Excel Driver (*.xls, *xlsx, *.xlsm, *.xlsb) 」 => 完成 => ODBC Microsoft Excel 設定 => 確定 => ODBC 資料來源管理員(64位元) => 確定



在Rgui/RStudio中 使用ODBC讀取 Excel 檔案 (Windows為例)

- STEP(2): Connect and import the data with ODBC

```
> install.packages("RODBC", repos = "http://cran.csie.ntu.edu.tw")
> library(RODBC)
> con <- odbcConnect('MyData-Excel')
> con
RODBC Connection 1
Details:
  case=nochange
  DSN=MyData-Excel
  DBQ=D:\MYR\Data\mydata.xlsx
  DefaultDir=D:\MYR\Data
  DriverId=1046
  FIL=excel 12.0
  MaxBufferSize=2048
  PageTimeout=5
> (test.data <- sqlFetch(con, '工作表1')) # returns a data frame object
Name Gender   Birthday Income           EventTime
1  John      M 1973-01-03  162.2 1899-12-30 13:00:00
2  Mary      F 1982-07-02   90.8 1899-12-30 23:50:00
3   Tim      M 1977-06-30   68.5 1899-12-30 02:30:00
4   Ron      M 1968-10-15  220.1 1899-12-30 05:20:00
5 Cathy      F 1980-12-01  150.0 1899-12-30 19:10:00
6   Sue      F 1976-04-02     NA 1899-12-30 12:00:00
> odbcClose(con)
```

```
> sqlTables(con)
                  TABLE_CAT TABLE_SCHEM TABLE_NAME   TABLE_TYPE REMARKS
1 D:\\MYR\\Data\\mydata.xlsx        <NA>  calculus$ SYSTEM TABLE    <NA>
2 D:\\MYR\\Data\\mydata.xlsx        <NA>   工作表1$ SYSTEM TABLE    <NA>
```

以ODBC連結其它資料來源

odbcConnect {RODBC}: ODBC Open Connections

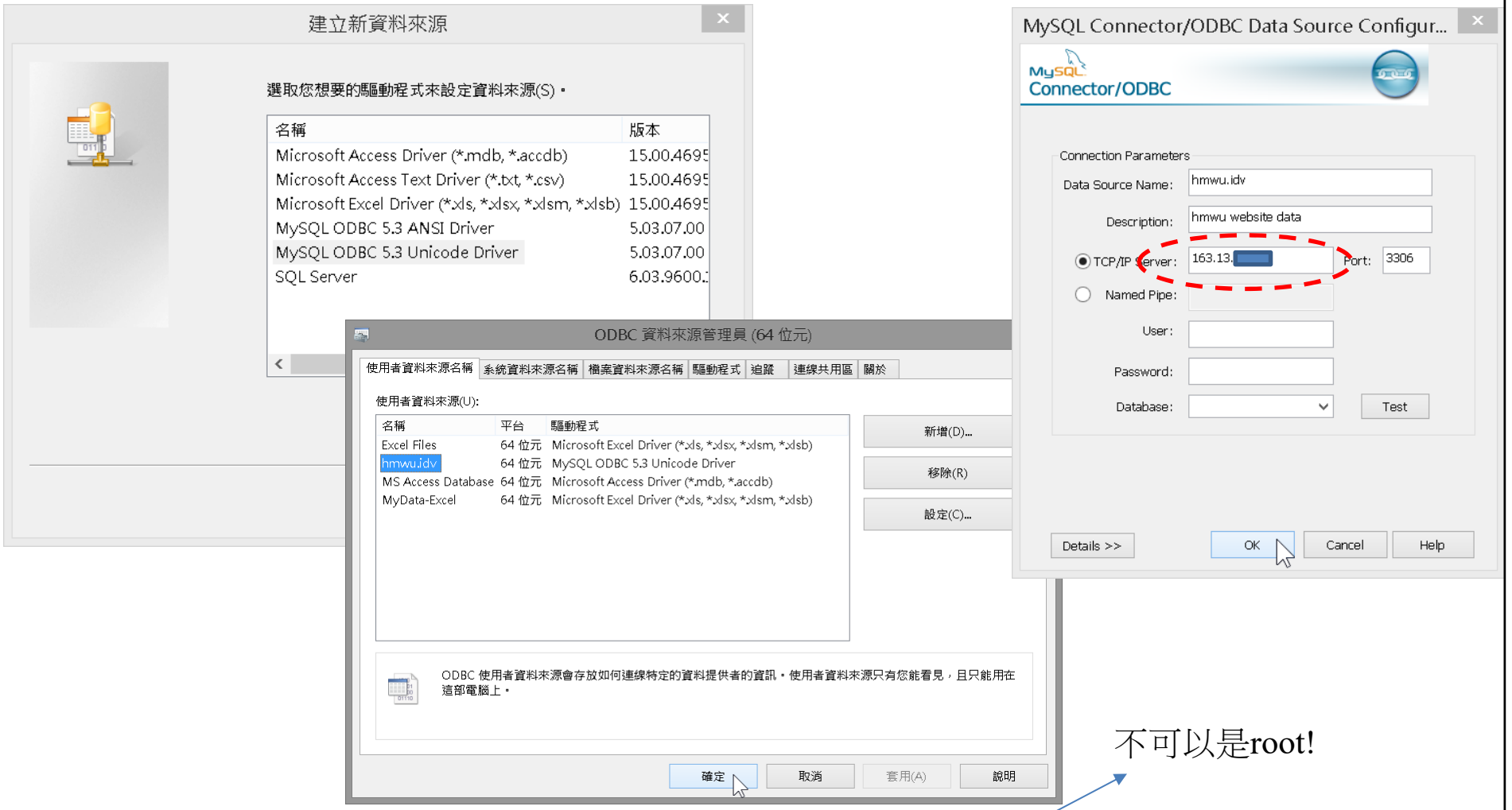
Description: Open connections to ODBC databases.

Usage:

```
odbcConnect(dsn, uid = "", pwd = "", ...)
odbcDriverConnect(connection = "", case, believeNRows = TRUE,
                  colQuote, tabQuote = colQuote,
                  interpretDot = TRUE, DBMSencoding = "",
                  rows_at_time = 100, readOnlyOptimize = FALSE)
odbcReConnect(channel, ...)
odbcConnectAccess(access.file, uid = "", pwd = "", ...)
odbcConnectAccess2007(access.file, uid = "", pwd = "", ...)
odbcConnectDbase(dbf.file, ...)
odbcConnectExcel(xls.file, readOnly = TRUE, ...)
odbcConnectExcel2007(xls.file, readOnly = TRUE, ...)
```

https://rviews.rstudio.com/2017/05/17/databases-using-r/
Databases using R

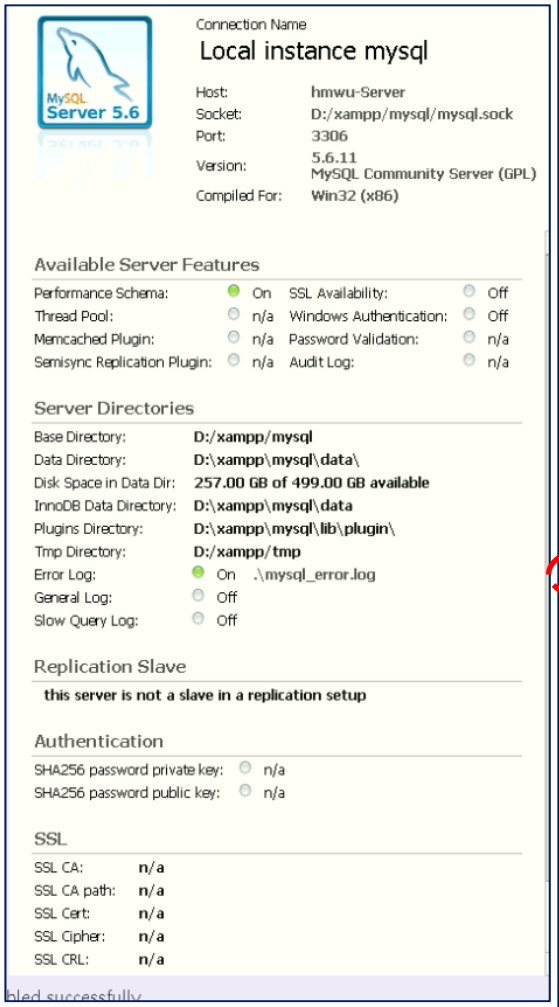
在Rgui/RStudio中 利用RODBC 與MySQL連線

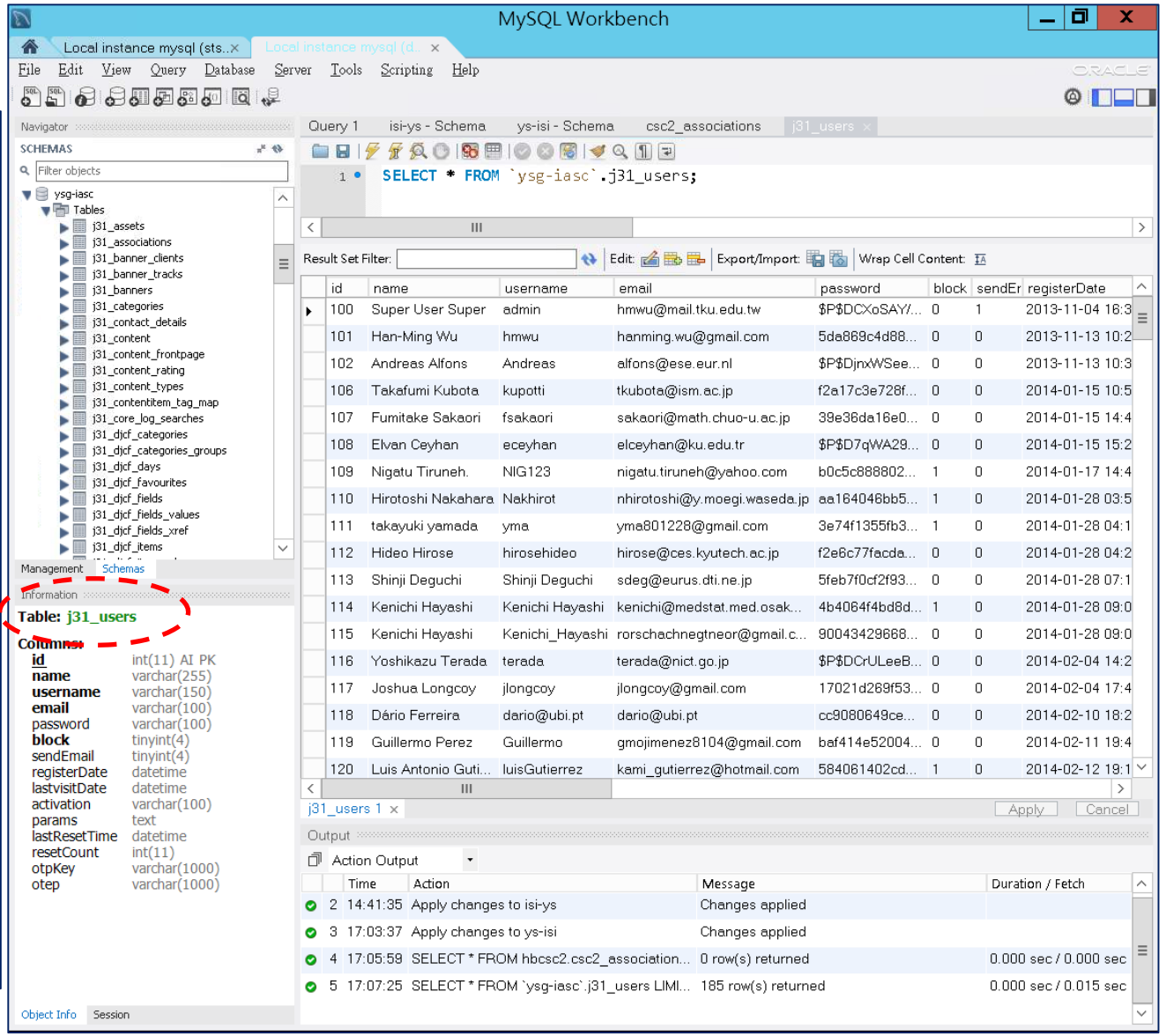


不可以是root!

```
> library(RODBC)
> con <- odbcConnect(dsn = 'hmwu.idv', uid = "hankwu", pwd = "xxxxx")
```

MySQL Server (windows為例)





在Rgui/RStudio中 利用RMySQL讀取MySQL資料庫的資料 (localhost)

RMySQL: Database Interface and 'MySQL' Driver for R

```
> library(DBI)
> library(gWidgets)
> library(RMySQL)
> library(dbConnect)
> con <- dbConnect(MySQL(), dbname = "ysg-iasc", host="localhost",
+                  username="root", password="xxxxxx")
> dbSendQuery(con,"SET NAMES utf8") #設定 UTF-8，避免中文亂碼
<MySQLResult:4065160,1,0>
> dbListTables(con)
 [1] "j31_assets"                    "j31_associations"
 [3] "j31_banner_clients"            "j31_banner_tracks"
...
[91] "j31_wf_profiles"               "j31_widgetkit_widget"
> data.users <- dbReadTable(con, "j31_users")
> class(data.users)
[1] "data.frame"
> head(data.users)
   id             name username                     email
1 100 Super User Super    admin      hmwu@mail.tku.edu.tw
2 101      Han-Ming Wu     hmwu      hanming.wu@gmail.com
3 102   Andreas Alfons  Andreas         alfons@ese.eur.nl
...
                                                  password block sendEmail
1                                $P$DCXoSAY/mf.s3mzaG9yQZr9NPd3pMX0     0         1
2 5da869c4d88338db86a5fa4e99723241:rBmCPPEczKD0SZG7krJeSNGQAekJavUV     0         0
...
```

登錄資訊可在「\mysql\bin\my.ini」或「mysql/bin/my.cnf」中新增一區段敘述[group]。

在Rgui/RStudio中 利用RMySQL讀取MySQL資料庫的資料 (localhost)

```
> dbListFields(con, "j31_users")
 [1] "id"       "name"           "username"       "email"          "password"
 [6] "block"    "sendEmail"      "registerDate"   "lastvisitDate" "activation"
[11] "params"   "lastResetTime" "resetCount"     "otpKey"         "otep"
> sel <- "SELECT name, email, sendEmail FROM j31_users" # 使用SQL語法讀取資料
> users.selected <- dbGetQuery(con, sel)
> head(users.selected)
              name                     email sendEmail
1 Super User Super      hmwu@mail.tku.edu.tw         1
2      Han-Ming Wu      hanming.wu@gmail.com         0
3   Andreas Alfons         alfons@ese.eur.nl         0
4  Takafumi Kubota          tkubota@ism.ac.jp         0
5 Fumitake Sakaori sakaori@math.chuo-u.ac.jp         0
6     Elvan Ceyhan        elceyhan@ku.edu.tr         0
> dbDisconnect(con)
[1] TRUE
```

dbWriteTable: data frame -> database table.

To retrieve results a chunk at a time, use **dbSendQuery**, **dbFetch**, then **dbClearResult**. If you want all the results (and they'll fit in memory) use **dbGetQuery** which sends, fetches and clears for you

MySQL Taiwan 台灣MySQL技術研究站
http://www.mysql.tw/
SQL SELECT語法整理
http://www.mysql.tw/#!/2014/05/sql-select.html

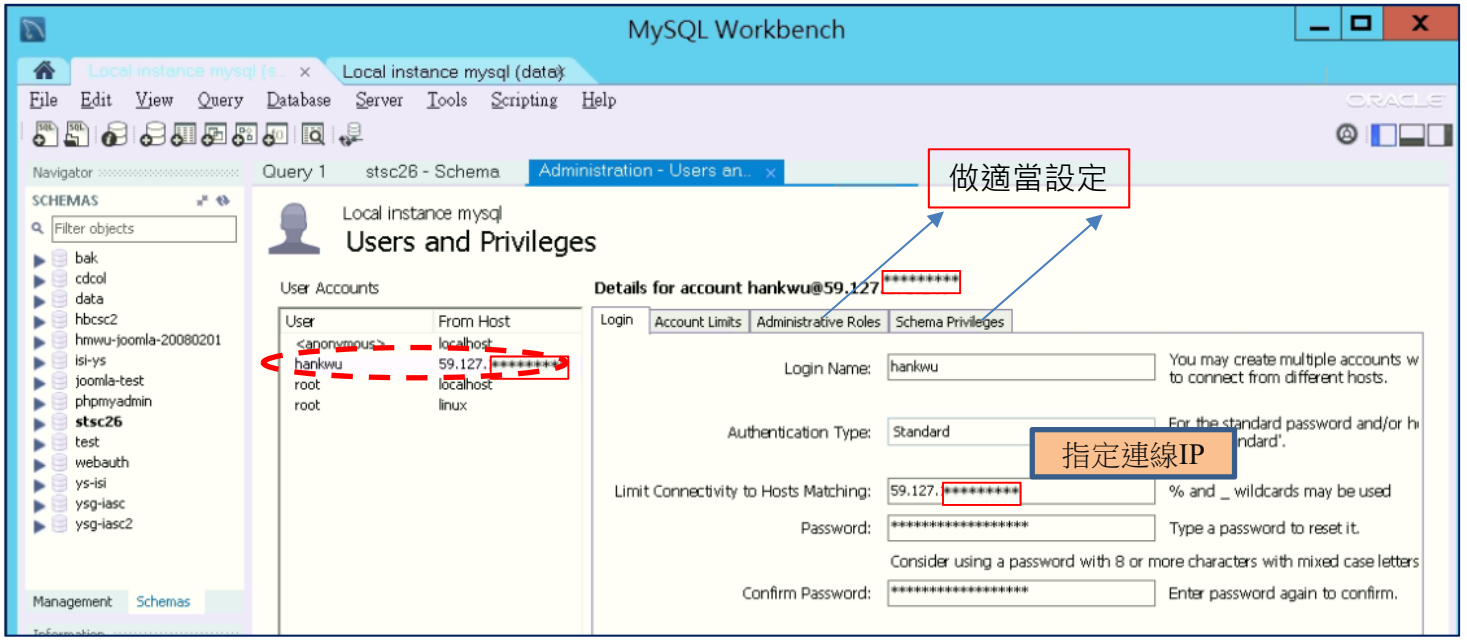
在Rgui/RStudio中 利用RMySQL讀取MySQL資料庫的資料 (remote host)

```
> con <- dbConnect(MySQL(), dbname = "ysg-iasc",
+                  username="root", password="xxxxxx",
+                  host="163.13.xxx.xxx", port=3306)
Error in .local(drv, ...) :
  Failed to connect to database: Error: Host '59-127-xxx-xxx.HINET-IP.hinet.net' is not
allowed to connect to this MySQL server
...
Error in .local(drv, ...) :
  Failed to connect to database: Error: Access denied for user 'root'@'59-127-xxx-
xxx.HINET-IP.hinet.net' (using password: YES)
```

MySQL 為了安全性的因素，禁止直接用 root 帳號從遠端連線。

```
con <- dbConnect(MySQL(), dbname = "ysg-iasc", username="hankwu", password="xxxxxx",
                 host="163.13.113.xxx", port=3306)
```

解決方式: 在remote host的MySQL中新增一名使用者，及設定其權限。



Memory Allocation in R

- 當R啟動時，設定最大可獲得的記憶體:

```
"C:\Program Files\R\R-3.2.2\bin\x64\Rgui.exe" --max-mem-size=2040M
```

  - 最小需求是32MB.
  - R啟動後僅可設定更高值，不能再用**memory.limit**設定較低的值。

```
> #目前使用的記憶體量
> memory.size(max = FALSE)
[1] 3845.87
>
> #從作業系統可得到的最大量記憶體
> memory.size(max = TRUE)
[1] 3846.25
>
> #列出目前記憶體的限制
> memory.limit(size = NA)
[1] 16343
>
> #設定新的記憶體限制為 1024 MB
> memory.limit(size = 1024)
[1] 16343
Warning message:
In memory.limit(size = 1024) : 無法減少記憶體限制：已忽略
```

- R與Windows作業系統

  (理論上)最大可獲得的記憶體

  - 32-bit R + 32-bit Windows: 2GB.
  - 32-bit R + 64-bit Windows: 4GB.
  - 64-bit R + 64-bit Windows: 8TB.

object.size(object)

Report the Space Allocated for an Object:

- 儲存R物件所佔用的記憶體估計。

```
object.size(x)
print(object.size(x), units = "Mb")
```

```
> n <- 10000
> p <- 200
> myData <- as.data.frame(matrix(rnorm(n*p), ncol = p, nrow=n))
> print(object.size(myData), units = "Mb")
15.3 Mb

> write.table(myData, "myData.txt") ## 約 34.7 MB

> InData <- read.table("myData.txt")
> print(object.size(InData), units = "Mb")
15.6 Mb
```

NOTE: Under any circumstances, you cannot have more than $2^{31}-1=2,147,483,647$ rows or columns.

變數標籤

```
> library(Hmisc)
> weight <- c(21, 65, 43)
> height <- c(164, 182, 170)
> label(weight) <- "體重"; label(height) <- "身高"
> units(weight) <- "公斤"; units(height) <- "公分"
> weight
體重 [公斤]
[1] 21 65 43
> height
身高 [公分]
[1] 164 182 170
> mydata <- data.frame(weight=weight, height=height)
> mydata
  weight height
1     21    164
2     65    182
3     43    170
```

```
> label(mydata)
weight height
"體重" "身高"
> # units(mydata) can't work
> # apply(mydata, 2, units) can't work
> lapply(mydata, units)
$weight
[1] "公斤"

$height
[1] "公分"
```

目錄下之檔案

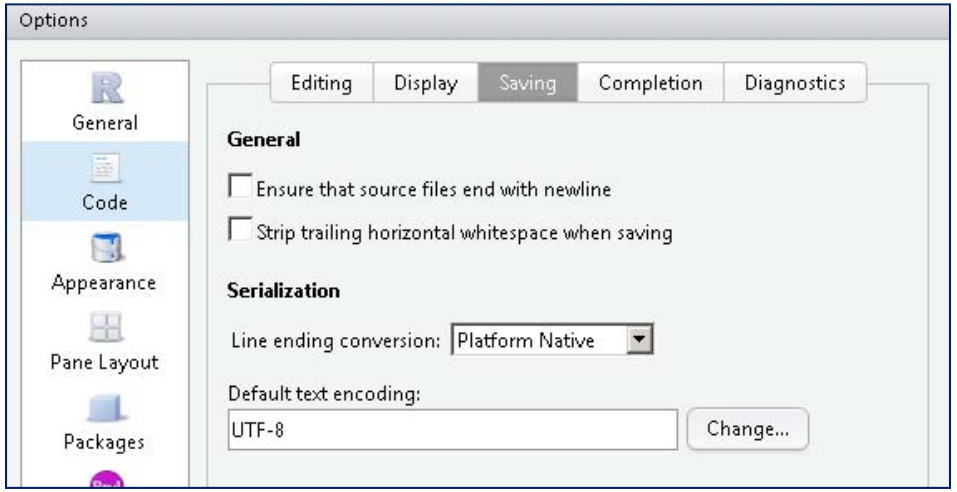
```
> getwd()
[1] "E:/08-MyProjects/07-graphics.SDA/MyPackage/graphics.SDA"
> list.dirs()
 [1] "."          "./.Rproj.user"
 [3] "./.Rproj.user/A3175805"        "./.Rproj.user/A3175805/ctx"
..
[57] "./src-i386"    "./src-x64"
> list.files() # dir()
 [1] "data"      "demo"      "DESCRIPTION"   "exploreSDA.dll"  "extdata"   "face-pairs.pdf"
 [7] "face-plot-index.pdf" "graphics.SDA.Rproj"  "inst"   "man"   "NAMESPACE"  "R"
[13] "raw-data"  "readme.txt"    "src"       "src-i386"  "src-x64"
> list.files(R.home())
 [1] "bin"  "CHANGES"  "COPYING"  "doc"  "etc"  "include"  "library"  "MD5"
 [9] "modules"  "README"   "README.R-3.4.0" "share"  "src"  "Tcl"  "tests"   "unins000.dat"
[17] "unins000.exe"
> dir("./data", pattern = "txt$")
[1] "3D_spatial_network.txt"  "city.txt"  "glass_214x9.txt"  "id.txt"
> file.info(dir())
                    size isdir mode               mtime               ctime               atime exe
data                   0  TRUE  777 2017-08-27 20:09:24 2015-05-03 21:29:23 2017-08-27 20:09:24  no
...
readme.txt          4052 FALSE  666 2015-05-17 11:11:56 2015-05-04 11:57:54 2016-09-11 09:08:03  no
src                    0  TRUE  777 2017-03-18 12:26:45 2015-05-04 21:50:53 2017-03-18 12:26:45  no
...
```

讀取資料含中文之編碼問題

- R & RStudio Troubleshooting Guide
  https://github.com/dspim/R/wiki/R-&-RStudio-Troubleshooting-Guide
- Mac/Linux系統預設格式是utf-8，Windows系統則是big-5（正體中文）。(必要時可在R之外進行轉碼後再讀檔)
- 指令中含有encoding之參數:

```
> source("myRcode.R", encoding = "utf-8")
> readLines("mydata.csv", encoding = "big5")
> read.table(..., fileEncoding = "", encoding = "unknown",...)
> data <- iconv(data, "big5", "utf8") # 將資料轉成 UTF-8
```

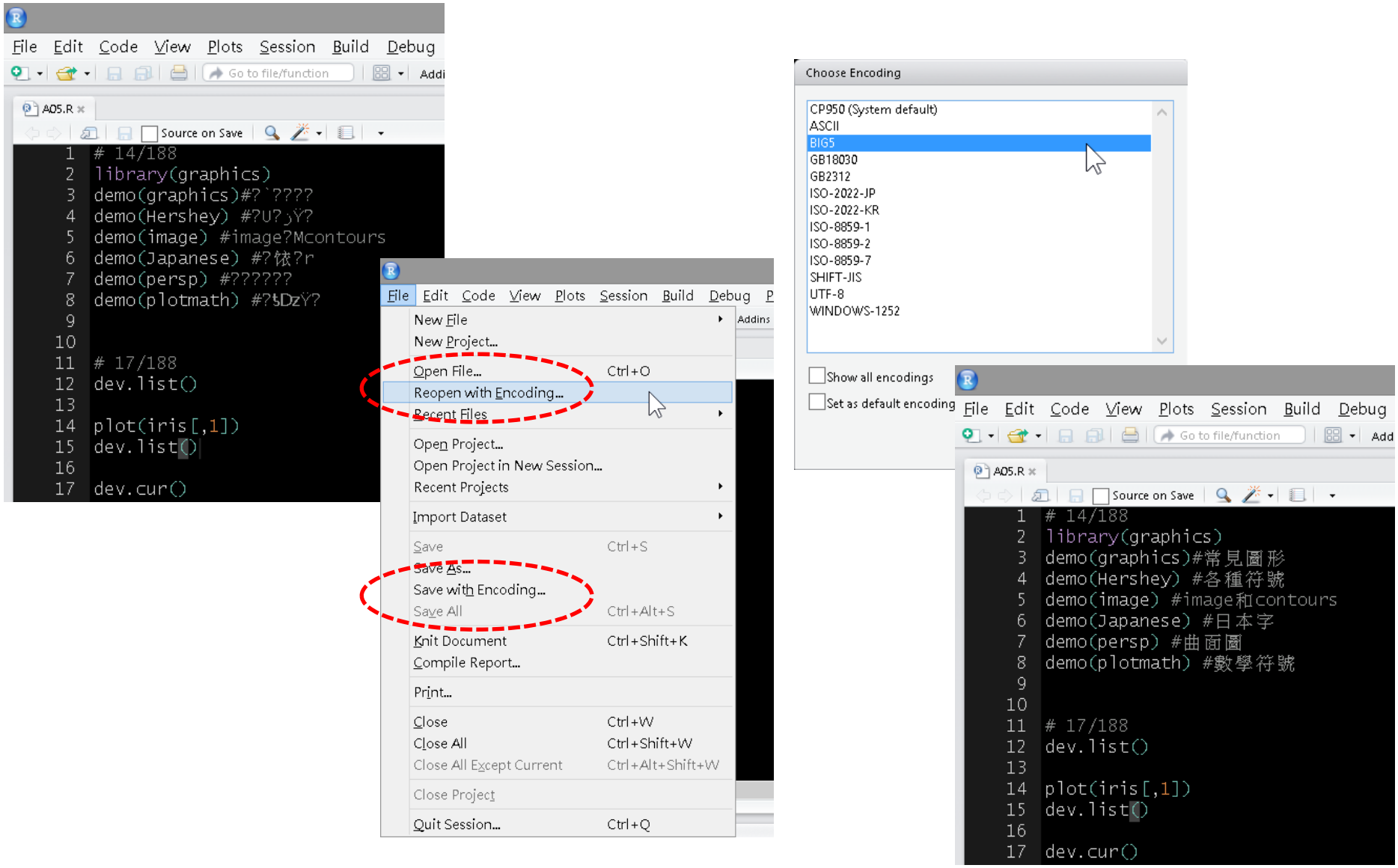
- R-Studio軟體編碼設定:



NOTE: 繪圖無法顯示中文?

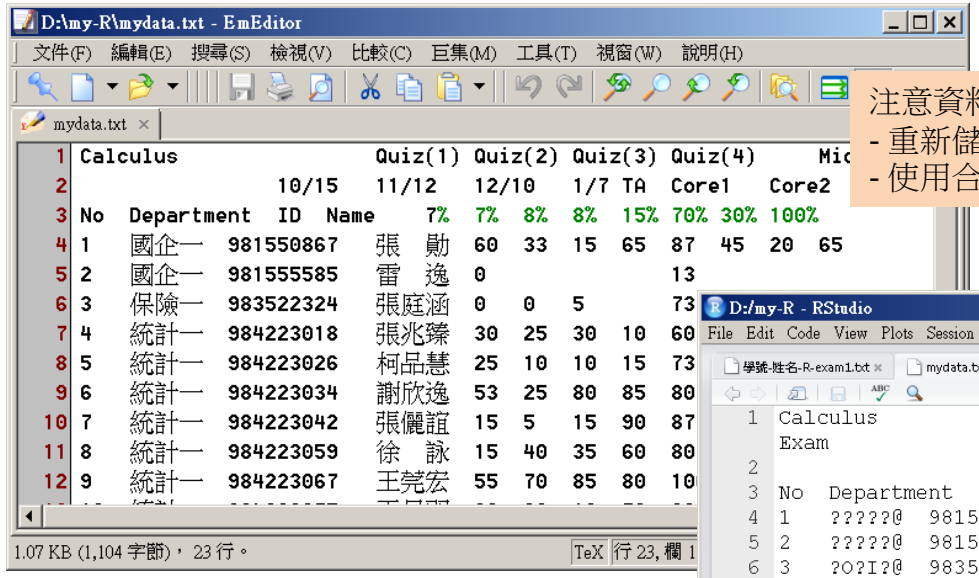
- Mac的預設字型serif沒有中文，需以par(family="STHeiti")重新設定字型。
- Note: Rmarkdown使用PostScript字形，以par重新設定可能還是無法正常顯示中文。
- **NOTE: 目錄不要是中文名。**

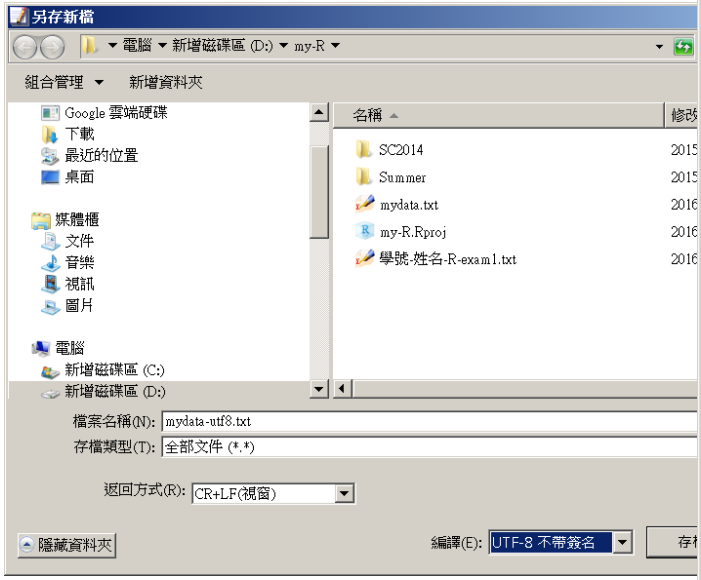
方法1: 利用RStudio的「Reopen with Encoding...」

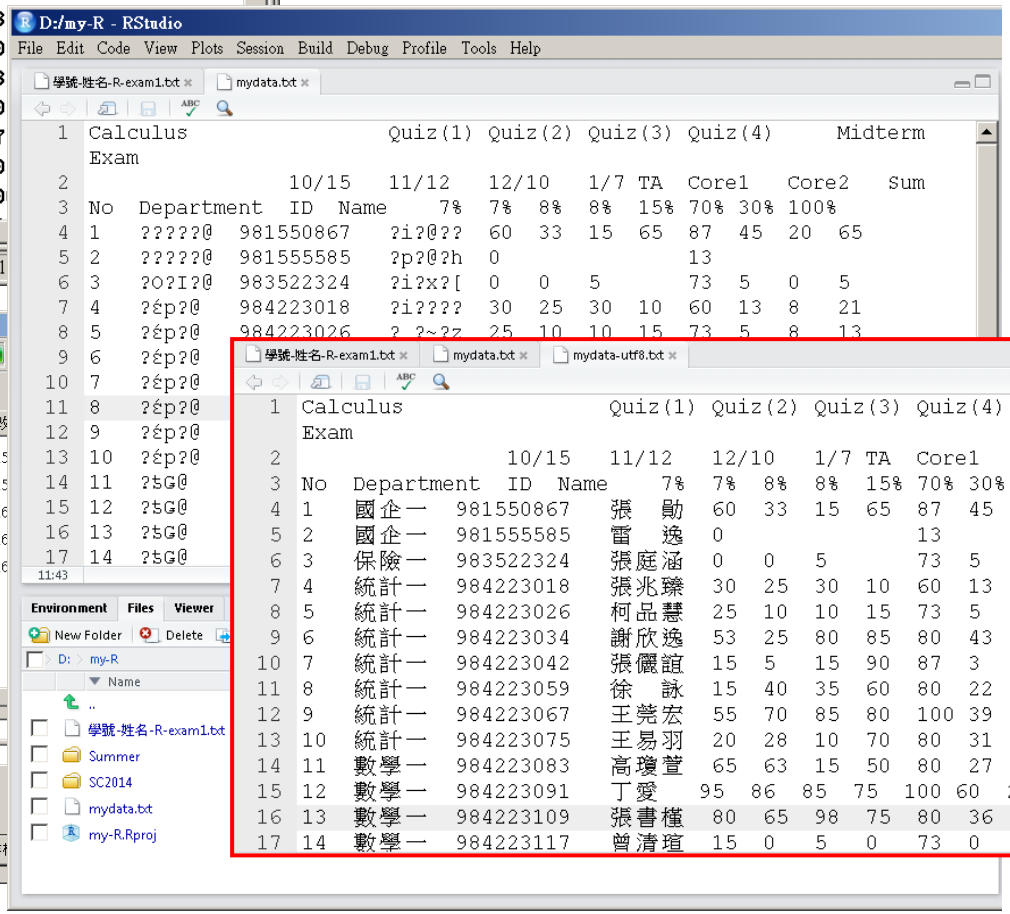


方法2: 將含中文之資料重新以UTF-8存檔，再載入RStudio

注意資料儲存之編碼為Big5、Utf-8、ANCI或其它。
- 重新儲存資料檔、編碼為Utf-8。
- 使用合適的編碼參數: `read.table("data.txt", encoding="anci")`







	Calculus Exam			Quiz(1)	Quiz(2)	Quiz(3)	Quiz(4)		Midterm		
			10/15	11/12	12/10	1/7 TA	Core1	Core2	Sum		
No	Department	ID	Name	7%	7%	8%	8%	15%	70%	30%	100%
1	國企一	981550867	張 勛	60	33	15	65	87	45	20	65
2	國企一	981555585	雷 逸	0				13			
3	保險一	983522324	張庭涵	0	0	5		73	5	0	5
4	統計一	984223018	張兆臻	30	25	30	10	60	13	8	21
5	統計一	984223026	柯品慧	25	10	10	15	73	5	8	13
6	統計一	984223034	謝欣逸	53	25	80	85	80	43	30	73
7	統計一	984223042	張儷誼	15	5	15	90	87	3	0	3
8	統計一	984223059	徐 詠	15	40	35	60	80	22	20	42
9	統計一	984223067	王莞宏	55	70	85	80	100	39	10	49
10	統計一	984223075	王易羽	20	28	10	70	80	31	5	36
11	數學一	984223083	高瓊萱	65	63	15	50	80	27	30	57
12	數學一	984223091	丁愛	95	86	85	75	100	60	20	80
13	數學一	984223109	張書樵	80	65	98	75	80	36	28	64
14	數學一	984223117	曾清瑄	15	0	5	0	73	0	7	7

讀中文資料檔編碼問題

```
> x <- c("曾寶儀", "蔡依琳", "吳瀞惠", "林志玲", "李伃晞")
> Encoding(x)
[1] "unknown" "unknown" "UTF-8"   "unknown" "UTF-8"
>
> getOption("encoding") # options(encoding="utf-8")
[1] "utf-8"
> options(stringsAsFactors = FALSE)
>
> (mydata1 <- read.table("NameAge1.txt", header = T, sep = "\t"))
    姓名 年紀
1 曾寶儀   12
2 蔡依琳   11
3 林志玲   23
>
> read.table("NameAge2.txt", header = T, sep="\t")
Error in scan(file = file, what = what, sep = sep, quote = quote, dec = dec,  :
  第 3 列沒有 2 個元素
...
> read.table("NameAge2.txt", header = T, sep="\t", fileEncoding = "utf8", encoding = "UTF-8")
Error in scan(file = file, what = what, sep = sep, quote = quote, dec = dec,  :
  第 3 列沒有 2 個元素
...
> read.csv("NameAge2.txt")
   姓名.年紀
1 曾寶儀\t12
2 蔡依琳\t11
3         吳
Warning messages:
1: In read.table(file = file, header = header, sep = sep, quote = quote,  :
  輸入連結 'NameAge2.txt' 中的輸入不正確
...
```

NameAge1.txt

姓名	年紀
曾寶儀	12
蔡依琳	11
林志玲	23

NameAge2.txt

姓名	年紀
曾寶儀	12
蔡依琳	11
吳瀞惠	34
林志玲	23
李伃晞	32

read.table {utils}, read.csv {utils}

See also: http://stackoverflow.com/questions/22876746/how-to-read-data-in-utf-8-format-in-r

讀中文資料檔編碼問題


```
> library(readr)
> (mydata2 <- read_delim("NameAge2.txt", delim="\t"))
Parsed with column specification:
cols(
  姓名 = col_character(),
  年紀 = col_integer()
)
# A tibble: 5 × 2
          姓名  年紀
         <chr> <int>
1        曾寶儀    12
2        蔡依琳    11
3 吳<U+701E>惠     34
4        林志玲    23
5 李<U+4F03>晞     32
> mydata2$姓名
[1] "曾寶儀" "蔡依琳" "吳瀞惠" "林志玲" "李伾晞"
> as.data.frame(mydata2)
          姓名 年紀
1        曾寶儀   12
2        蔡依琳   11
3 吳<U+701E>惠    34
4        林志玲   23
5 李<U+4F03>晞    32
>
> Encoding(mydata2[[1]])
[1] "UTF-8" "UTF-8" "UTF-8" "UTF-8" "UTF-8"
> enc2native(mydata2[[1]])
[1] "曾寶儀"       "蔡依琳"       "吳<U+701E>惠" "林志玲"       "李<U+4F03>晞"
> enc2utf8(mydata2[[1]])
[1] "曾寶儀" "蔡依琳" "吳瀞惠" "林志玲" "李伾晞"
```

```
> str(mydata2)
Classes ‘tbl_df’, ‘tbl’ and 'data.frame':       5 obs. of  2 variables:
 $ 姓名: chr  "曾寶儀" "蔡依琳" "吳<U+701E>惠""| __truncated__ "林志玲" ...
 $ 年紀: int  12 11 34 23 32
 - attr(*, "spec")=List of 2
  ..$ cols   :List of 2
  .. ..$ 姓名: list()
  .. .. ..- attr(*, "class")= chr  "collector_character" "collector"
  .. ..$ 年紀: list()
  .. .. ..- attr(*, "class")= chr  "collector_integer" "collector"
  ..$ default: list()
  .. ..- attr(*, "class")= chr  "collector_guess" "collector"
  ..- attr(*, "class")= chr "col_spec"
```

```
> c(mydata2)[[1]]
[1] "曾寶儀" "蔡依琳" "吳瀞惠" "林志玲" "李伾晞"
> apply(mydata2, 2, c)  # try apply(mydata2, 2, enc2utf8)
     姓名       年紀
[1,] "曾寶儀" "12"
[2,] "蔡依琳" "11"
[3,] "吳瀞惠" "34"
[4,] "林志玲" "23"
[5,] "李伾晞" "32"
```

讀中文資料檔編碼問題

Sys.setlocale(category = "LC_ALL", locale = "cht")

```
WARNING: Failed to restore workspace from 'E:/10-R/01-ä¸»é¡Œ/A03-
Graphics&Visualization/åœ°åœ–/maps/.RData'
Reason: cannot open the connection
> getwd()
[1] "E:/10-R/01-主題/A03-Graphics&Visualization/地圖/maps"
Warning messages:
1: In dir.create(tempPath, recursive = TRUE) :
  cannot create dir 'E:\10-R\01-??', reason 'Invalid argument'
2: In readChar(con, 5L, useBytes = TRUE) :
  cannot open compressed file 'E:/10-R/01-??/A03-Graphics&Visualization/??/maps/.RData',
probable reason 'Invalid argument'
> Sys.setlocale(category = "LC_ALL", locale = "cht")
[1] "LC_COLLATE=Chinese (Traditional)_Taiwan.950;LC_CTYPE=Chinese
(Traditional)_Taiwan.950;LC_MONETARY=Chinese
(Traditional)_Taiwan.950;LC_NUMERIC=C;LC_TIME=Chinese (Traditional)_Taiwan.950"
> getwd()
[1] "E:/10-R/01-主題/A03-Graphics&Visualization/地圖/maps"
```

R 讀取、處理、輸出 UTF-8 萬國碼資料教學與範例
https://officeguide.cc/r-read-process-write-utf8-data-tutorial-examples/?fbclid=IwAR0eY2m5I_z-6wRdI4Z-rgMokmijtKzegK3SxS4cW63rjZ4Z5P-536y-vUA

讀中文資料檔編碼問題

Sys.setlocale(category = "LC_ALL", locale = "cht")

```
> Xinbei <- st_read("201807/Xinbei.shp", options = "ENCODING=UTF-8", stringsAsFactors = FALSE)
> head(Xinbei, 3)
...
  U_ID      CODEBASE        CODE1    CODE2  TOWN_ID                                        TOWN COUNTY_ID
1 2293 A6515-0078-00 A6515-05-009 A6515-05 65000150  <U+00A4><U+00AD><U+00AA><U+0470><cf>     65000
2 2294 A6515-0079-00 A6515-05-010 A6515-05 65000150  <U+00A4><U+00AD><U+00AA><U+0470><cf>     65000
3 2295 A6517-0046-00 A6517-03-001 A6517-03 65000170        <U+00AA>L<U+00A4>f<U+00B0><cf>     65000
>
> Xinbei$TOWN <- iconv(Xinbei$TOWN, to="UTF-8")
> head(Xinbei)
...
  U_ID      CODEBASE        CODE1    CODE2  TOWN_ID   TOWN COUNTY_ID
1 2293 A6515-0078-00 A6515-05-009 A6515-05 65000150 ¤ªÑ°Ï     65000
2 2294 A6515-0079-00 A6515-05-010 A6515-05 65000150 ¤ªÑ°Ï     65000
3 2295 A6517-0046-00 A6517-03-001 A6517-03 65000170 ªL¤f°Ï     65000
> Sys.setlocale(category = "LC_ALL", locale = "cht")
[1] "LC_COLLATE=Chinese (Traditional)_Taiwan.950;LC_CTYPE=Chinese
(Traditional)_Taiwan.950;LC_MONETARY=Chinese
(Traditional)_Taiwan.950;LC_NUMERIC=C;LC_TIME=Chinese (Traditional)_Taiwan.950"
>
> Xinbei <- st_read("201807/Xinbei.shp", options = "ENCODING=UTF-8", stringsAsFactors = FALSE)
> head(Xinbei)
> Xinbei$TOWN <- iconv(Xinbei$TOWN, to="UTF-8")
> head(Xinbei)
...
  U_ID      CODEBASE        CODE1    CODE2  TOWN_ID   TOWN COUNTY_ID
1 2293 A6515-0078-00 A6515-05-009 A6515-05 65000150 五股區     65000
2 2294 A6515-0079-00 A6515-05-010 A6515-05 65000150 五股區     65000
3 2295 A6517-0046-00 A6517-03-001 A6517-03 65000170 林口區     65000
```

其它相關套件

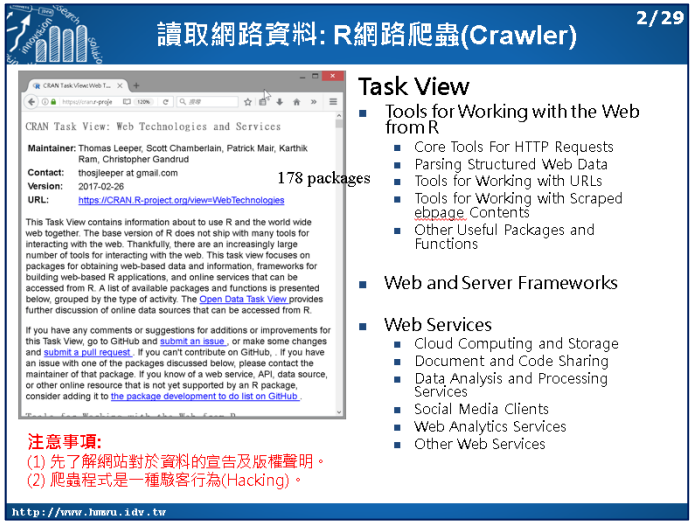
- **arrow**: A package for reading and writing Arrow files.
- **avro**: A package for reading and writing data in Apache Avro format.
- **data.table**: A package that can read and manipulate large data sets efficiently.
- **DBI**: A database interface definition for communication between R and relational database management systems.
- **feather**: A lightweight binary file format for storing data frames that can be read and written by multiple programming languages, including R.
- **haven**: A package that can read and write data files in various formats including SAS, SPSS and Stata files.
- **hdf5r**: A package for reading and writing HDF5 files.
- **httr**: A package for working with web APIs and downloading data from web-based sources.
- **openxlsx**: A package that can read and write Excel files in both .xlsx and .xls formats.
- **readr**: A package for reading delimited text files such as CSV and TSV files.
- **RJDBC**: A package for connecting to databases using JDBC connections.
- **RPostgreSQL**: A package for connecting to PostgreSQL databases.
- **rio**: A package that import and export streamlined data.
- **rvest**: A package for web scraping and extracting data from HTML pages.
- **XLConnect**: Provides comprehensive functionality to read, write and format Excel data.

讀取資料相關講義

D01
讀取大型資料in R
吳漢銘
國立臺北大學 統計學系
http://www.hmwu.idv.tw

大綱
2/43
記憶體設置、物件大小、計算執行(資料讀取)時間
Handling Large Data Sets in R
讀取目錄下符合目標的(多個)檔案資料: list.files
直接讀取壓縮檔(zip)內之檔案
讀取HTML網頁表格，讀取XML表格
讀取影像檔案
從資料庫(MySQL)讀取資料
GREA: read ALL the data into R/Importing Data with RStudio
讀取部份資料進入R計算(readbulk)
fread {data.table}: Fast and friendly file finagler
讀取檔案部份欄位資料
如何讓read.table讀較大的資料速度更快
http://www.hmwu.idv.tw/index.php/r-software
http://www.hmwu.idv.tw

D02
R網路爬蟲
吳漢銘
國立臺北大學 統計學系
http://www.hmwu.idv.tw


讀取網路資料: R網路爬蟲(Crawler)
2/29
CRAN Task View: Web Technologies and Services
Maintainer: Thomas Leeper, Scott Chamberlain, Patrick Mair, Karthik Ram, Christopher Gandrud
Contact: thosjleeper at gmail.com
Version: 2017-02-26
URL: https://CRAN.R-project.org/view=WebTechnologies
178 packages
Task View
Tools for Working with the Web from R
Core Tools For HTTP Requests
Parsing Structured Web Data
Tools for Working with URLs
Tools for Working with Scraped ebpage Contents
Other Useful Packages and Functions
Web and Server Frameworks
Web Services
Cloud Computing and Storage
Document and Code Sharing
Data Analysis and Processing Services
Social Media Clients
Web Analytics Services
Other Web Services
注意事項:
(1) 先了解網站對於資料的宣告及版權聲明。
(2) 爬蟲程式是一種駭客行為(Hacking)。
http://www.hmwu.idv.tw