데이터 조작하기

dplyr을 중심으로

유충현

Updated: 2019/12/26

Overview

1. General Information
2. Read Large Data
3. Data Wrangling
4. SQL Based Data Wrangling
5. dplyr Data Wrangling
6. Parallel Processing

General Information

서론

외부 데이터를 읽어들여 데이터 프레임을 만들고, 이를 조작하여 데이터 분석을 위한 데이터 셋을 생성하는 것은 데이터 분석 과정에서 대단히 중요한 과정이다.

R에서 데이터를 조작하는 기술은 데이터 조작 함수를 구사하는 능력에 비례한다. base, stats 패키지에 포함된 전통적인 R(S-PLUS) 데이터 조작 함수의 대체제는 sqldf 패키지를 거쳐, dplyr 패키지로 꽃을 피웠다 할 수 있다.

왜 이렇게 느린가?

"내가 다룰 데이터는 iris와 USArrests 데이터 프레임이 전부인 줄 알았다. 그런데, 이제는 bigdata라는 용어가 판치고 있다."

Data Size에 비례하지 않는 Speed

작은 용량의 데이터에 최적화된 read.*() 함수군은 대용량 데이터의 입력에 견디기 힘들 정도의 수행 속도를 보임.
수행 속도는 데이터의 양에 선형증가가 아닌, 임계치를 넘으면 기하급수적으로 속도가 둔화됨.

심플한 방법은 없을까?

"나는 데이터를 조작하기 위해서 그 많은 연산자와 함수를 익혀야만 하였다. 그리고 시간이 흘러 여러 권법보다는 하나의 절대 비기가 필요함을 깨달았다."

당신 또한 이 많은 기능을 익혀야할 수도 있음

- operators:
  - `extract or replace, [, [[, $, @.`
- functions:
  - `apply: apply, lapply, mapply, tapply, sweep.`
  - `subset or transform: subset, transform, within.`
  - `summary: table, aggregate, xtabs.`

놀고 먹게 둘 수 없어!!!

"어느 정도 코딩을 할 줄 알고부터는 프로그램이 아닌 CPU와 메모리가 눈에 들어오고, 그들의 태만이 결딜 수 없이 싫었다."

그들에게 일을 시키기 위해서 우리는 채찍을 들어야 함

- memory:
  - `in-memory`.
    - 데이터를 메모리에 올려야되서 용량의 한계 직면
    - 해결: ff, bigmemory 등의 패키지의 이용
    - garbage collection, gc() 함수 활용
- CPU:
  - `single core processing`.
    - 특정 core만 혹사하고, 나머지는 놀고 일을 하지 않음.
    - 해결: multi core processing, "백짓장도 맞들면 낮다."
  - parellel, doMc, doParallel, doSNOW, ... 패키지 이용.

나의 작업환경

```
               _
platform       x86_64-apple-darwin15.6.0
arch           x86_64
os             darwin15.6.0
system         x86_64, darwin15.6.0
status
major          3
minor          5.2
year           2018
month          12
day            20
svn rev        75870
language       R
version.string R version 3.5.2 (2018-12-20)
nickname       Eggshell Igloo
```

데이터 준비하기

- en.openfoodfacts.org.products.tsv (963MB) 다운로드 후,
- FoodFacts.tsv라는 이름으로 변경 - 탭 구분자 (tab delimeter)
- https://www.kaggle.com/openfoodfacts/world-food-facts

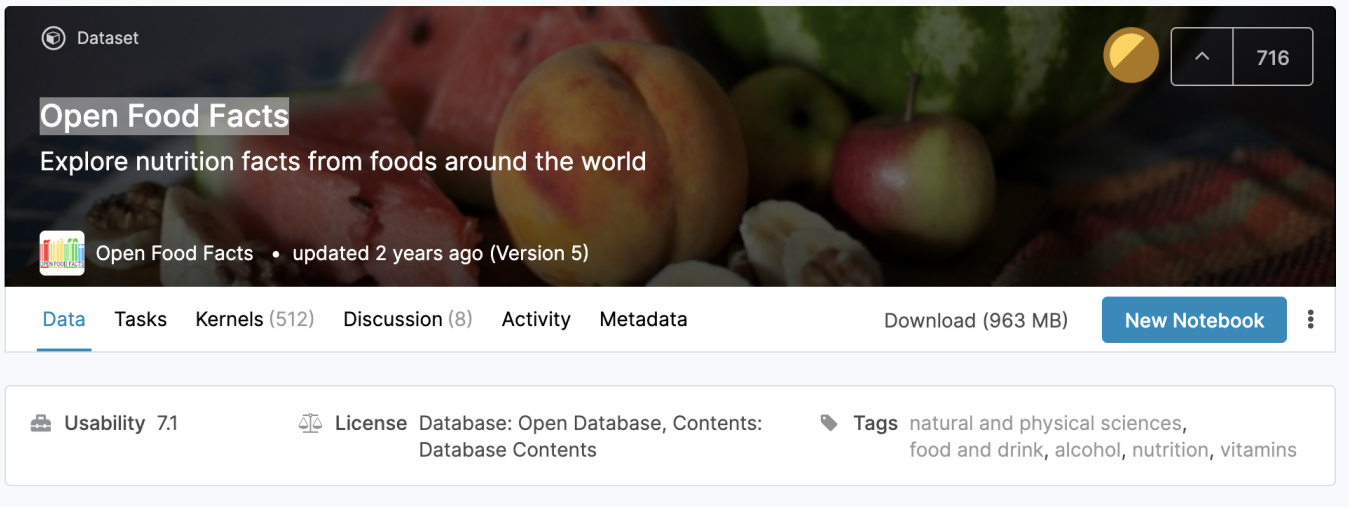


그림: Kaggle 홈페이지의 다운로드 화면

Read Large Data

저용량 데이터 파일 읽기

read.csv 함수는 대표적인 데이터 입력 함수임

```
> fname <- "./data/iris.csv"
> write.csv(iris, file = fname, row.names = FALSE)
> system.time(iris2 <- read.csv(fname))

   user  system elapsed
  0.002   0.001   0.002

> head(iris2, n = 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
```

대용량 데이터 파일 읽기 1

read.csv 함수는 대용량 데이터 입력에 적합치 않음

```
> fname <- "./data/FoodFacts.tsv"
> file.size(fname) / 1024^2

[1] 963.456

> system.time(foodfact <- read.csv(fname, header = TRUE,
+                                  sep = "\t"))

   user  system elapsed
276.966   6.849 284.059

> format(object.size(foodfact), units = "Mb")

[1] "980 Mb"
```

대용량 데이터 파일 읽기 2

read_csv, read_tsv 함수는 대용량 데이터 입력에 적합함

```
> library(readr)
>
> system.time(foodfact2 <- read_tsv(fname))

   user  system elapsed
  7.747   0.355   8.126

> format(object.size(foodfact2), units = "Mb")

[1] "938.7 Mb"
```

read.csv vs read_csv

read.csv 함수는 data.frame 클래스 객체를 만듦

```
> class(foodfact)

[1] "data.frame"

> dim(foodfact)

[1] 248523    163

> head(foodfact[, c(1, 3:4)], n = 1)

  code                      creator  created_t
1 3087 openfoodfacts-contributors 1474103866
```

read.csv vs read_csv

read_csv, read_tsv 함수는 tbl_df 클래스 객체를 만듦

```
> class(foodfact2); dim(foodfact2)

[1] "spec_tbl_df" "tbl_df"      "tbl"         "data.frame"
[1] 314496    163

> head(foodfact2[, c(1, 3:4)], n = 2)

# A tibble: 2 x 3
  code          creator                       created_t
  <chr>         <chr>                             <dbl>
1 0000000003087 openfoodfacts-contributors 1474103866
2 0000000004530 usda-ndb-import            1489069957
```

read.csv vs read_csv

read.csv와 read_csv의 목적은 같지만 상이한 함수임

표: read.csv vs read_csv (read_tsv)

구분	**read.csv**	**read_csv**
package	utils	readr
family functions	read.*()	read_*()
output	data.frame	tbl_df
describe column type	colClasses	col_types

describe column type

column type	read.csv	read_csv
character	"character"	"c"
integer	"integer"	"i"
number	없음	"n"
double	"double"	"d"
logical	"logical"	"l"
factor	"factor"	없음
date	"Date"	"D"
date time	"POSIXct"	"T"
time	없음	"t"
guess (default)	없음	"?"
skip the column	NULL	"_" 혹은 "-"

예제 데이터 파일 만들기

FoodFacts.csv 파일의 1, 3, 4 컬럼을 갖는 4건의 데이터 추출

```
> cmd <- "head -n 4 ./data/FoodFacts.tsv"
> cmd <- paste(cmd, "| cut -f 1,3-4 -d '\t'")
> cmd <- paste(cmd, "> ./data/FoodFacts2.csv")
> system(cmd)
>
> system("cat ./data/FoodFacts2.csv", intern = TRUE)

[1] "code\tcreator\tcreated_t"
[2] "0000000003087\topenfoodfacts-contributors\t1474103866"
[3] "0000000004530\tusda-ndb-import\t1489069957"
[4] "0000000004559\tusda-ndb-import\t1489069957"
```

read.csv 사용하기

nrows 인수와 colClasses 인수 사용하기

```
> fname <- "./data/FoodFacts2.csv"
> read.csv(fname, nrows = 1, sep = "\t")

  code                      creator  created_t
1 3087 openfoodfacts-contributors 1474103866

> ctype <- c("character", "character", "double")
> read.csv(fname, nrows = 2, colClasses = ctype, sep = "\t")

           code                      creator  created_t
1 0000000003087 openfoodfacts-contributors 1474103866
2 0000000004530           usda-ndb-import 1489069957
```

read_csv 사용하기

n_max 인수와 col_types 인수 사용하기

```
> read_tsv(fname, n_max = 1)

# A tibble: 1 x 3
  code          creator                       created_t
  <chr>         <chr>                             <dbl>
1 0000000003087 openfoodfacts-contributors 1474103866

> read_tsv(fname, n_max = 1, col_types = c("ncd"))

# A tibble: 1 x 3
   code creator                       created_t
  <dbl> <chr>                             <dbl>
1  3087 openfoodfacts-contributors 1474103866
```

실습 (Question) : CSV 파일 읽기

"./data/FoodFacts.tsv 파일로 다음을 수행하세요."

1. 파일을 read.csv() 함수로 읽어 food1 데이터 프레임에 할당하세요.
2. 파일을 read_tsv() 함수로 읽어 food2 tbl_df에 할당하세요.
3. 두 객체의 데이터 건수, 변수 건수를 비교해 보세요.
4. 두 가지 방법의 수행 속도를 비교해 보세요.
5. 두 객체의 앞 세건의 데이터를 살펴보세요. 어떤 차이가 있나요?

실습 (Answer) : CSV 파일 읽기

```
> library(readr)
> fname <- "./data/FoodFacts.tsv"
>
> # question 1, 4
> system.time(food1 <- read.csv(fname, sep = "\t"))
> system.time(food2 <- read_tsv(fname))
>
> # question 3
> dim(food1)
> dim(food2)
>
> # question 5
> food1[1:3, ]
> food2[1:3, ]
```

Data Wrangling

extract operator

extract variables

- 데이터 프레임에서 변수(열)를 선택하는 것
  - data.frame[, index]
  - data.frame[, variable names]

extract obervations

- 데이터 프레임에서 특정 관측치(행)를 선택하는 것
  - data.frame[index,]
  - data.frame[row names,]
  - data.frame[condition,]

exercises: extract variables

```
> USArrests[1:2, c(1:2, 4)]                          # index

         Murder Assault Rape
Alabama    13.2     236 21.2
Alaska     10.0     263 44.5

> USArrests[1:2, c("Murder", "Assault", "Rape")] # names

         Murder Assault Rape
Alabama    13.2     236 21.2
Alaska     10.0     263 44.5
```

exercises: extract obervations

```
> USArrests[1:2, ]                    # index

         Murder Assault UrbanPop Rape
Alabama    13.2     236       58 21.2
Alaska     10.0     263       48 44.5

> USArrests[c("Nevada", "Texas"), ] # row names

       Murder Assault UrbanPop Rape
Nevada   12.2     252       81 46.0
Texas    12.7     201       80 25.5
```

exercises: extract obervations

```
> (cond <- USArrests$Rape > 45)

 [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[10] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[19] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[28]  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[46] FALSE FALSE FALSE FALSE FALSE

> USArrests[cond, ]  # condition

       Murder Assault UrbanPop Rape
Nevada   12.2     252       81   46
```

exercises: subset function

```
> subset(USArrests, select = c(Murder, Rape),
+        subset = Rape > 40)

           Murder Rape
Alaska       10.0 44.5
California    9.0 40.6
Nevada       12.2 46.0
```

실습 (Question) : extract data frame

"USArrests 데이터 프레임으로 다음을 수행하세요."

1. Murder 변수가 평균 이상인 관측치를 추출하세요.
2. "M"으로 시작하는 주의 Assault, UrbanPop 변수를 추출하세요.
    - 힌트) row.names() 함수, grep() 함수 응용

실습 (Answer) : extract data frame

```
> # questtion 1
> USArrests[USArrests$Murder >= mean(USArrests$Murder), ]
>
> # questtion 2
> grep("^M", rownames(USArrests))
> grep("^M", rownames(USArrests), value = TRUE)
> USArrests[grep("^M", rownames(USArrests)),
+           c("Assault", "UrbanPop")]
```

transformation

- 데이터 프레임의 변수를 가공하여 새 변수 생성
  - transform function
  - within function

```
> air <- airquality[1:3, ]
> air

  Ozone Solar.R Wind Temp Month Day
1    41     190  7.4   67     5   1
2    36     118  8.0   72     5   2
3    12     149 12.6   74     5   3
```

exercises: transform function

transform은 함수의 인수값으로 파생변수를 지정함

```
> transform(air, lOzone = log(Ozone),
+           Temp = round((Temp-32) / 1.8, 1))

  Ozone Solar.R Wind Temp Month Day   lOzone
1    41     190  7.4 19.4     5   1 3.713572
2    36     118  8.0 22.2     5   2 3.583519
3    12     149 12.6 23.3     5   3 2.484907
```

exercises: within function

within는 expr 인수값의 블록 안에 파생변수를 지정함

```
> within(air, expr = {
+   lOzone <- log(Ozone)
+   cTemp <- round((Temp-32) / 1.8, 1)
+   rm(Temp)
+ })

  Ozone Solar.R Wind Month Day cTemp   lOzone
1    41     190  7.4     5   1  19.4 3.713572
2    36     118  8.0     5   2  22.2 3.583519
3    12     149 12.6     5   3  23.3 2.484907
```

summarise data

contingency tables

- 범주형 데이터 조합의 frequency 구하기
  - table function
  - ftable function
  - xtabs function

grouped summary

- 범주형 데이터 조합별로 연속변수 집계
  - aggregate function
  - tapply function

exercises: contingency tables - 1

```
> table(mtcars$cyl)                    # 1-dimension


 4  6  8
11  7 14

> table(mtcars$cyl, mtcars$gear)       # 2-dimension


    3  4  5
  4 1  8  2
  6 2  4  1
  8 12 0  2
```

exercises: contingency tables - 2

```
> ftable(mtcars[c("cyl", "am", "gear")])

        gear  3  4  5
cyl am
4   0         1  2  0
    1         0  6  2
6   0         2  2  0
    1         0  2  1
8   0        12  0  0
    1         0  0  2
```

exercises: contingency tables - 3

```
> xtabs(~ cyl + gear, data = mtcars,
+       subset = am == 1)

   gear
cyl 4 5
  4 6 2
  6 2 1
  8 0 2
```

실습 (Question) : contingency tables

"USArrests 데이터 프레임으로 다음을 수행하세요."

1. within() 함수로 Murder 변수의 소수점을 제거한 Murder2 변수를 추가하여 USA라는 이름의 데이터 프레임을 만들어라.
2. USA 데이터 프렘임의 3건을 화면에 출력하세요.
   - 힌트) head() 함수 혹은 "[" 연산자 이용
3. Murder2 변수별로 주의 개수를 카운트해 보세요.

실습 (Answer): contingency tables

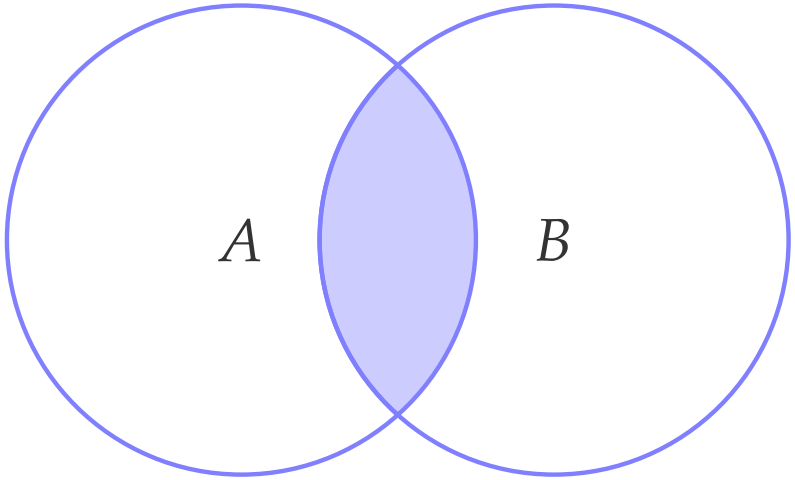
```
> # question 1
> USA <- within(USArrests,
+               expr = {Murder2 <- round(Murder)})
>
> # question 2
> head(USA, n = 3)
>
> # question 3
> table(USA[, "Murder2"])
```

merge two data frames

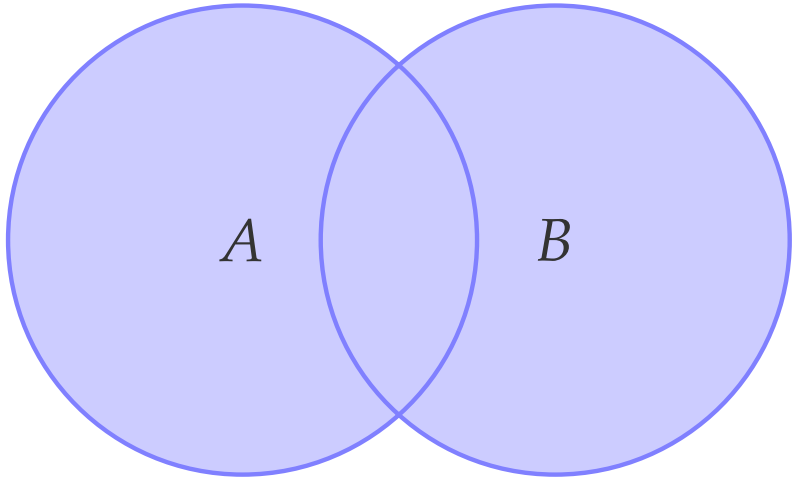
- 동일한 key를 기준으로 두 데이터 프레임 결합
- merge 함수를 사용하나, 속도가 느림
- combine(join) 유형
  - inner join
    - 동일 key 값을 갖는 A, B 교집합만 취함
  - outer join
    - key 값 기준 합집합을 취함
  - left outer join
    - 모든 A와 동일 key 값의 B를 취함
  - right outer join
    - 동일 key 값의 A와 모든 B를 취함

combine data frames - venn diagram

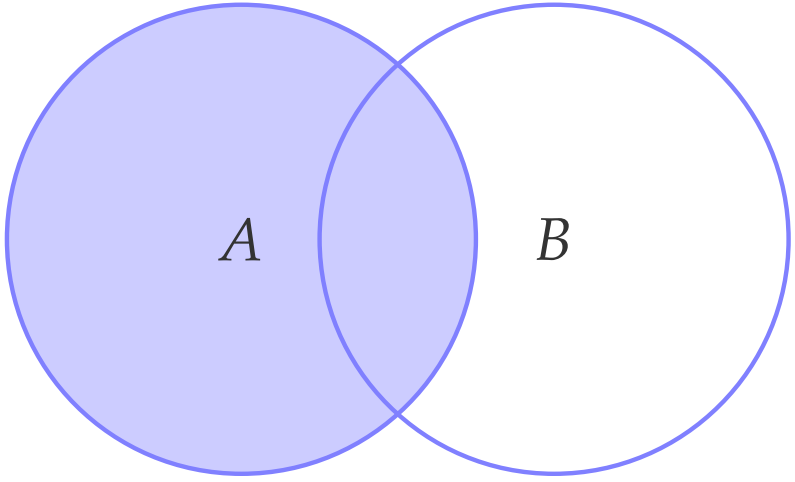
$A \cap B$: inner join



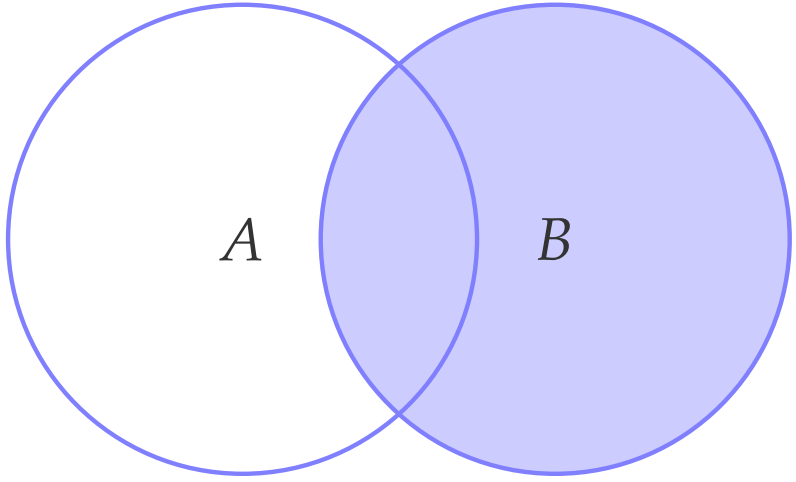
$A \cup B$: outer join



left outer join



right outer join



exercises: data 생성

```
> bid <- c("001", "002", "003")
> bname <- c("R Visualization", "Introduce R", "R world")
> books <- data.frame(bid = bid, bname = bname)
> books

  bid           bname
1 001 R Visualization
2 002     Introduce R
3 003         R world
```

exercises: data 생성

```
> bid <- c("002", "003", "004")
> auth <- c("Lee", "Kim", "Ryu")
> price <- c(32000, 45000, 28000)
> bookinfo <- data.frame(bid = bid, auth = auth,
+                        price = price)
> bookinfo

  bid auth price
1 002  Lee 32000
2 003  Kim 45000
3 004  Ryu 28000
```

exercises: inner/outer join

```
> merge(books, bookinfo)                  # inner join

  bid       bname auth price
1 002 Introduce R  Lee 32000
2 003     R world  Kim 45000

> merge(books, bookinfo, all = TRUE) # outer join

  bid           bname auth price
1 001 R Visualization <NA>    NA
2 002     Introduce R  Lee 32000
3 003         R world  Kim 45000
4 004           <NA>  Ryu 28000
```

exercises: outer join

```
> merge(books, bookinfo, all.x = TRUE) # left outer join

  bid           bname auth price
1 001 R Visualization <NA>    NA
2 002     Introduce R  Lee 32000
3 003         R world  Kim 45000

> merge(books, bookinfo, all.y = TRUE) # right outer join

  bid       bname auth price
1 002 Introduce R  Lee 32000
2 003     R world  Kim 45000
3 004        <NA>  Ryu 28000
```

실습 (Question) : combine data frames

"USArrests 데이터 프레임, state.x77 행렬로 다음을 수행하세요."

1. USArrests의 변수 이름과 state.x77의 열 이름을 조회하세요.
2. merge(USArrests, state.x77)를 수행해 보세요.
    - 어떤 결과가 수행되었는지 설명하세요.
3. merge(USArrests, state.x77, by = 0)를 수행해 보세요.
    - 어떤 결과가 수행되었는지 설명하세요.

실습 (Answer) : combine data frames

```
> # question 1
> names(USArrests)
> colnames(state.x77)
>
> # question 2
> merge(USArrests, state.x77)
>
> # question 3
> merge(USArrests, state.x77, by = 0)
```

SQL Based Data Wrangling

sqldf package

sqldf package 소개

- SQL 구문으로 data frames을 조작
- in-memory DBMS인 SQLite를 통합하여 구현
- 작은 사이즈의 데이터는 오히려 수행속도가 늦음
  - 데이터를 메모리(SQLite DBMS)에 올리는 오버로드 존재

sqldf package 학습

- sqldf Github page
  - `https://github.com/ggrothendieck/sqldf`
- SQLite Documents page
  - `https://sqlite.org/doclist.html`

sqldf package 개념

data fram을 SQLite에 올려 DB Connection으로 SQL 질의 수행

그림: sqldf 개념도

SQL의 구조

Structured Query Language 구조

표: SQL 구조

Data Wrangling	SQL	비고
subset variables	select 절	컬럼명 기술
data transformation	select 절	파생컬럼 정의
subset observations	where 절	논리식 정의
group data	group by 절	집계 key 정의
summarise Data	select 절	집계식 정의
data sorting	order by 절	sort key 정의
combine data set	join 구문	join key 정의

exercises: sqldf - 1

```
> library(sqldf)
>
> sql <- "select `Petal.Length`, `Petal.Width`, Species"
> sql <- paste(sql, "from iris")
> sql <- paste(sql, "where `Sepal.Length` < 4.5")
> sqldf(sql)

  Petal.Length Petal.Width Species
1          1.4         0.2  setosa
2          1.1         0.1  setosa
3          1.3         0.2  setosa
4          1.3         0.2  setosa
```

exercises: sqldf - 2

```
> sql <- ""
> sql <- paste(sql, "select cyl")
> sql <- paste(sql, "      ,avg(hp) as mean_hp")
> sql <- paste(sql, "      ,count(*) as cnt")
> sql <- paste(sql, "  from mtcars")
> sql <- paste(sql, " where mpg > 20")
> sql <- paste(sql, " group by cyl")
> sqldf(sql)

  cyl   mean_hp cnt
1   4  82.63636  11
2   6 110.00000   3
```

exercises: sqldf - inner join

```
> sql <- ""
> sql <- paste(sql, "select x.bid, x.bname")
> sql <- paste(sql, "      ,y.auth, y.price")
> sql <- paste(sql, "from books x")
> sql <- paste(sql, "  inner join bookinfo y")
> sql <- paste(sql, "  on x.bid = y.bid")
>
> sqldf(sql)

  bid       bname auth price
1 002 Introduce R  Lee 32000
2 003     R world  Kim 45000
```

exercises: sqldf - left outer join

```
> sql <- ""
> sql <- paste(sql, "select x.bid, x.bname")
> sql <- paste(sql, "      ,y.auth, y.price")
> sql <- paste(sql, "from books x")
> sql <- paste(sql, "  left outer join bookinfo y")
> sql <- paste(sql, "  on x.bid = y.bid")
>
> sqldf(sql)

  bid           bname auth price
1 001 R Visualization <NA>    NA
2 002     Introduce R  Lee 32000
3 003         R world  Kim 45000
```

exercises: sqldf - outer join

- sqldf는 outer join을 지원하지 않음

```
> sql <- ""
> sql <- paste(sql, "select x.bid, x.bname")
> sql <- paste(sql, "      ,y.auth, y.price")
> sql <- paste(sql, "from books x outer join bookinfo y")
> sql <- paste(sql, "on x.bid = y.bid")
> sqldf(sql)

Error: RIGHT and FULL OUTER JOINs are not currently
supported
```

dplyr Data Wrangling

dplyr 함수

dplyr 패키지는 함수를 중심으로 데이터 조작을 수행함

표: dplyr 대표 함수

Data Wrangling	Functions
subset variables	select
subset observations	filter, distinct, top_n, sample_n, ...
data transformation	mutate, transmute, mutate_each, ...
group data	group_by
summarise data	summarise, summarise_each, count
data sorting	arrange
combine data set	inner_join, lef_join, right_join, full_join

subset variables

- select: 변수 이름을 지정하여 해당 변수만 추출

```
> library(dplyr)
> iris2 <- iris[1:3, ]
> select(iris2, Sepal.Length, Species)

  Sepal.Length Species
1          5.1  setosa
2          4.9  setosa
3          4.7  setosa
```

subset variables

- select: tbl_df 클래스

```
> class(foodfact2)

[1] "spec_tbl_df" "tbl_df"      "tbl"         "data.frame"

> select(foodfact2, code, product_name)[1:2, ]

# A tibble: 2 x 2
  code          product_name
  <chr>         <chr>
1 0000000003087 Farine de blé noir
2 0000000004530 Banana Chips Sweetened (Whole)
```

subset observations

- filter: 논리적 기준에 맞는 행 추출

```
> filter(iris, Sepal.Length > 7.6)

  Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
1          7.7         3.8          6.7         2.2 virginica
2          7.7         2.6          6.9         2.3 virginica
3          7.7         2.8          6.7         2.0 virginica
4          7.9         3.8          6.4         2.0 virginica
5          7.7         3.0          6.1         2.3 virginica
```

subset observations

- top_n: 상(하)위 n위(rank)의 행 추출
- sample_n: 무작위로 n 개 행 선택

```
> top_n(iris, 1, Sepal.Length)

  Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
1          7.9         3.8          6.4           2 virginica

> set.seed(1)
> sample_n(iris, 2)

  Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
1          5.1         3.4          1.5         0.2     setosa
2          5.7         2.8          4.5         1.3 versicolor
```

data transformation

- mutate: 하나 이상의 새로운 열을 계산 후 추가

```
> (air <- airquality[1:2, c("Ozone", "Wind", "Temp")])

  Ozone Wind Temp
1    41  7.4   67
2    36  8.0   72

> mutate(air, CTemp = round((Temp - 32) / 1.8, 1))

  Ozone Wind Temp CTemp
1    41  7.4   67  19.4
2    36  8.0   72  22.2
```

data transformation

- mutate_each: 모든 열에 특정 함수를 적용
- transmute: 하나 이상의 새로운 열을 계산, 원래 열은 삭제

```
> mutate_each(air, funs(log))

     Ozone     Wind     Temp
1 3.713572 2.001480 4.204693
2 3.583519 2.079442 4.276666

> transmute(air, CTemp = round((Temp - 32) / 1.8, 1))

  CTemp
1  19.4
2  22.2
```

group data

- group_by: 동일한 값을 갖는 행으로 데이터 그룹화

```
> group_by(mtcars[1:4, ], gear, cyl)

# A tibble: 4 x 11
# Groups:   gear, cyl [3]
    mpg   cyl  disp    hp  drat    wt  qsec    vs    am
* <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1  21       6   160   110  3.9   2.62  16.5     0     1
2  21       6   160   110  3.9   2.88  17.0     0     1
3  22.8     4   108    93  3.85  2.32  18.6     1     1
4  21.4     6   258   110  3.08  3.22  19.4     1     0
# . . .  with 2 more variables: gear <dbl>, carb <dbl>
```

summarise data

- summarise: 데이터를 단일 행 값으로 요약, group_by와 결합하여 사용

```
> iris2 <- group_by(iris, Species)
> summarise(iris2, avg = mean(Sepal.Width))

# A tibble: 3 x 2
  Species      avg
  <fct>      <dbl>
1 setosa      3.43
2 versicolor  2.77
3 virginica   2.97
```

summarise data

- summarise_each: 각 컬럼에 요약 기능 적용

```
> iris2 <- group_by(iris, Species)
> summarise_each(iris2, funs(mean))

# A tibble: 3 x 5
  Species  Sepal.Length Sepal.Width Petal.Length Petal.Width
  <fct>           <dbl>       <dbl>        <dbl>       <dbl>
1 setosa           5.01        3.43         1.46       0.246
2 versico. . .      5.94        2.77         4.26       1.33
3 virgini. . .      6.59        2.97         5.55       2.03
```

summarise data

- count: 각 수준별 행의 수(frequency) 계산

```
> count(iris, Species)

# A tibble: 3 x 2
  Species        n
  <fct>      <int>
1 setosa        50
2 versicolor    50
3 virginica     50
```

summarise data

- arrange: 열 값으로 행을 정렬 (기본-오름차순)

```
> iris2 <- iris[1:2, 1:4]
> arrange(iris2, Sepal.Width)

  Sepal.Length Sepal.Width Petal.Length Petal.Width
1          4.9         3.0          1.4         0.2
2          5.1         3.5          1.4         0.2

> arrange(iris2, desc(Sepal.Width))

  Sepal.Length Sepal.Width Petal.Length Petal.Width
1          5.1         3.5          1.4         0.2
2          4.9         3.0          1.4         0.2
```

combine data set

- inner_join: inner join

```
> inner_join(books, bookinfo)

  bid       bname auth price
1 002 Introduce R  Lee 32000
2 003     R world  Kim 45000
```

combine data set

- full_join: full outer join

```
> full_join(books, bookinfo)

  bid            bname auth price
1 001 R Visualization <NA>    NA
2 002     Introduce R  Lee 32000
3 003         R world  Kim 45000
4 004           <NA>  Ryu 28000
```

combine data set

- left_join: left outer join

```
> left_join(books, bookinfo)

  bid            bname auth price
1 001 R Visualization <NA>    NA
2 002     Introduce R  Lee 32000
3 003         R world  Kim 45000
```

combine data set

- right_join: right outer join

```
> right_join(books, bookinfo)

  bid       bname auth price
1 002 Introduce R  Lee 32000
2 003     R world  Kim 45000
3 004        <NA>  Ryu 28000
```

실습 (Question) : dplyr

"dplyr 패키지로 mtcars 데이터 프레임에 대해 다음을 수행하세요."

1. mpg, cyl, disp, hp 변수를 추출하여 mcar를 생성하세요.
2. mcar에서 row.names를 읽어서 carname라는 변수로 추가하세요.
3. mcar에서 cyl가 6 이상인 건을 추출하여 mcar2를 생성하세요.
4. mcar2 mpg 기준으로 상위 top 5를 추출하세요.
5. mcar2를 cyl, hp 별로 disp의 최대값, mpg의 평균을 구하세요.
    - 주문1 : group_by() 함수로 집계 기준을 정한 후 mcar3를 만들고,
    - 주문2 : mcar3에 summarise() 함수를 적용하되,
    - 주문3 : max.disp, mean.mpg 변수에 각각 최대값과 평균을 넣어라.

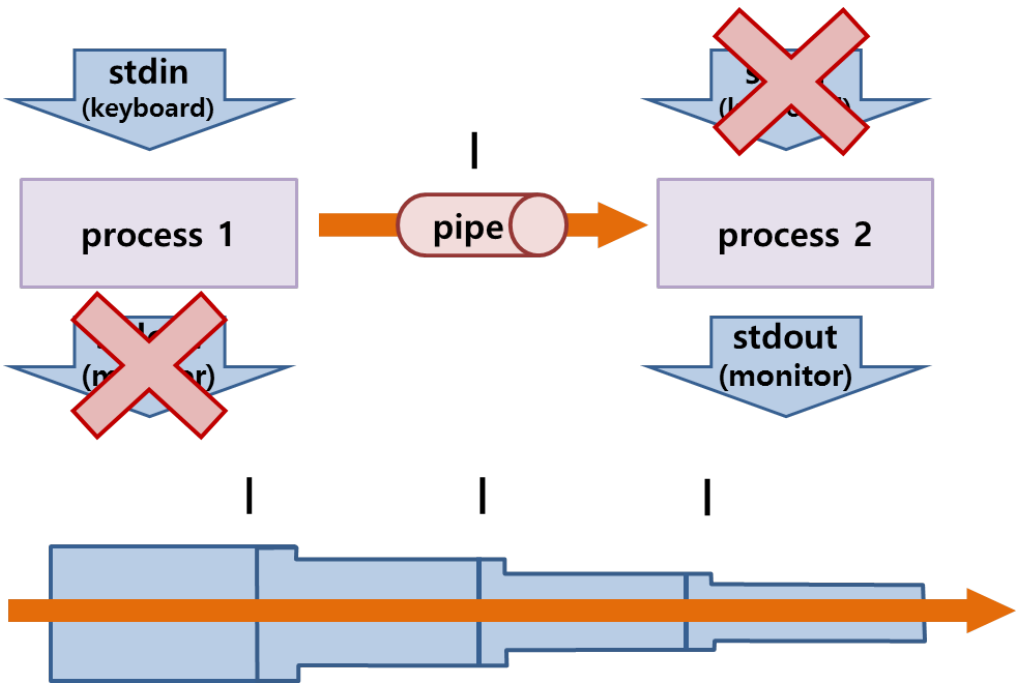
실습 (Answer) : dplyr

```
> # question 1
> mcar <- select(mtcars, mpg, cyl, disp, hp)
> # question 2
> mcar <- mutate(mcar, carname = row.names(mcars))
> mcar
> # question 3
> mcar2 <- filter(mcars, cyl >= 6)
> mcar2
> # question 4
> top_n(mcar2, 5, mpg)
> # question 5
> mcars3 <- group_by(mcars2, cyl, hp)
> summarise(mcars3, max.disp = max(disp),
+           mean.mpg = mean(mpg))
```

pipeline을 아시나요?

pipeline

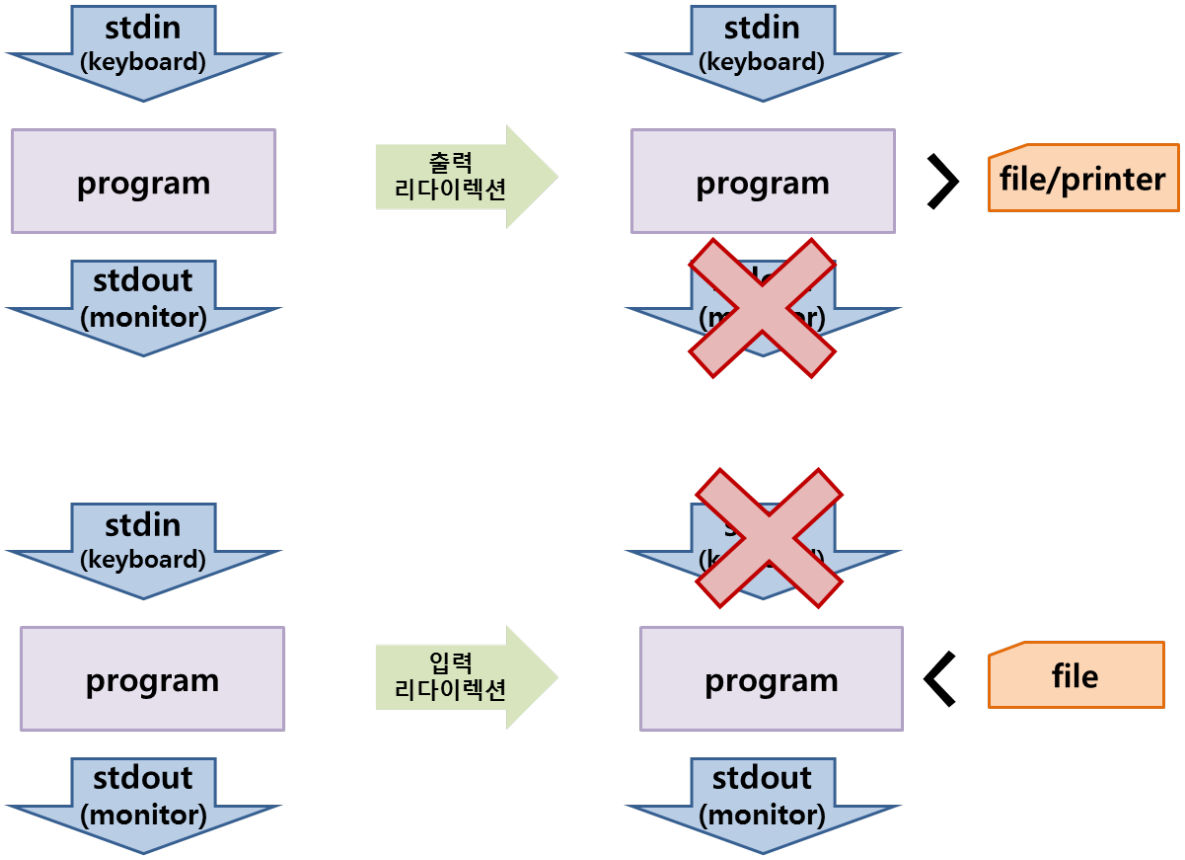
- 둘 이상의 명령어를 하나로 묶어,
- 앞 명령의 출력을 다음 명령의 입력으로 사용.
- 코드가 간결해지고 interaction 프로그램 가능.



redirection을 아시나요?

redirection

- 표준 입력이나 출력 방향을 재지향(redirection)



exercises: UNIX에서의 pipeline

FoodFacts.csv 앞 4건을 추출 후, 컴마를 기준으로 1, 3, 4번째 컬럼을 취해 FoodFacts2.tsv에 저장

```
# head -n 4 FoodFacts.tsv | cut -f 1,3-4 -d '\t'> FoodFacts2.tsv
```

rusr가 포함된 UNIX 사용자의 디스크 사용량 조회

```
# cat /etc/passwd | grep "/rusr" |cut -d: -f6 | xargs du -sh
24G     /home/rusr01
299M    /home/rusr02
3.0M    /home/rusr03
```

R에서의 pipeline

R 환경에서의 pipeline

- 최근 R 프로그램의 trend
- magrittr package : dplyr package도 상속함

- forward-pipe operator : lhs %>% rhs
  - rhs의 첫번째 인수로 lhs 사용
    - x %>% f(y) : f(x, y)
  - rhs 호출의 (첫번째 인수와) 다른 위치에 lhs 배치
    - y %>% f(x, .) : f(x, y)
  - 보조 목적으로 도트(.) 사용
    - iris %>% subset(1:nrow(.) %% 2 == 0)
- exposition-pipe operator : lhs %$% rhs
  - lhs에있는 이름(names)을 rhs 표현식에 사용
    - data.frame(z = rnorm(100)) %$% plot(z)

exercises: R에서의 pipeline

```
> library(magrittr)
> iris %>%
+   subset(Species == "setosa") %>%
+   .[, -5] %>%
+   apply(2, mean)

Sepal.Length  Sepal.Width Petal.Length  Petal.Width
       5.006        3.428        1.462        0.246

> iris %>%
+   subset(Sepal.Length > mean(Sepal.Length)) %$%
+   cor(Sepal.Length, Sepal.Width)

[1] 0.3361992
```

execrises: dplyr & magrittr

"품종별로 꽃받침 폭의 평균을 구한 후, 내림차순으로 정렬하라."

```
> iris %>%
+  group_by(Species) %>%
+  summarise(avg = mean(Sepal.Width)) %>%
+  arrange(desc(avg))

# A tibble: 3 x 2
  Species      avg
  <fct>      <dbl>
1 setosa      3.43
2 virginica   2.97
3 versicolor  2.77
```

execrises: dplyr & magrittr

"airquality에서 기온이 섭씨 30도 이상 풍속이 초속 4.5미터 이하의 월별 평균 오존농도를 구하라."

온도 변환

$$temp(^{\circ}C) = \frac{temp(^{\circ}F) - 32}{1.8}$$

풍속 변환

$$speed(m/s) = \frac{speed(miles/h) * 1609.34}{60 \times 60}$$

execrises: dplyr & magrittr

```
> airquality %>%
+   mutate(CTemp = round((Temp - 32) / 1.8, 1),
+          MWind = round(Wind * 1609.34 / 60^2, 1)) %>%
+   filter(CTemp >= 30, MWind <= 4.5) %>%
+   group_by(Month) %>%
+  summarise(AOzone = round(mean(Ozone, na.rm = TRUE))) %>%
+   filter(!is.na(AOzone))

# A tibble: 3 x 2
  Month AOzone
  <int>  <dbl>
1     7     81
2     8     90
3     9     77
```

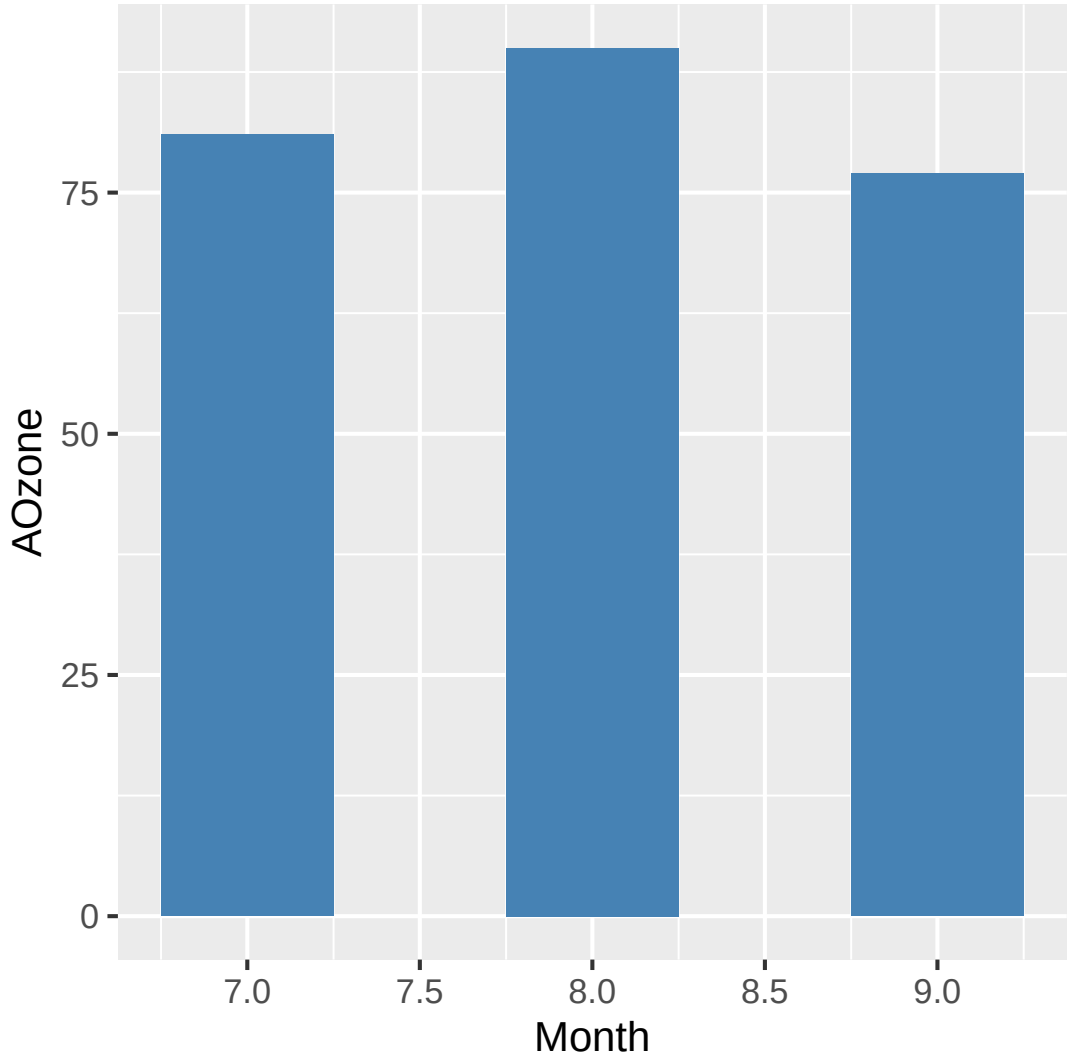
execrises: redirection - ggplot2와 찰떡 궁합

"magrittr를 이용해서 데이터 조작 결과를 바로 ggplot2로 보내서 시각화 수행"

```
> library(ggplot2)
> airquality %>%
+   mutate(CTemp = round((Temp - 32) / 1.8, 1),
+          MWind = round(Wind * 1609.34 / 60^2, 1)) %>%
+   filter(CTemp >= 30, MWind <= 4.5) %>%
+   group_by(Month) %>%
+  summarise(AOzone = round(mean(Ozone, na.rm = TRUE))) %>%
+   filter(!is.na(AOzone)) %>%
+   ggplot(aes(x = Month, y = AOzone)) +
+   geom_bar(stat = "identity", width = 0.5,
+            fill = "steelblue")
```

execrises: ggplot2와 찰떡 궁합

"ggplot을 연계하여 바로 시각화를 수행한 결과"



실습 (Question) : dplyr & magrittr

"dplyr, ggplot2 패키지로 mtcars 데이터 프레임에 대해
파이프라인을 이용해서 다음을 수행하세요."

1. 다음의 주문대로 계산하여 결과를 출력하세요.
    - mpg, cyl, disp, hp 변수를 추출한 후,
    - 행 이름 앞 단어로 브랜드를 구해 brand 변수를 추가한 후,
        - 힌트) sapply(strsplit(row.names(.), split = " "), "[", 1)
    - cyl가 6 이상인 건으로 필터링한 후,
    - brand별로 disp의 최대값과 mpg의 평균을 출력하세요.
2. 앞의 결과에 파이프라인을 연결하여 산점도를 그려보세요.
    - x-축은 disp의 최대값, y-축은 mpg의 평균
    - geom_smooth() 함수로 lm 추세선을 표현
    - 점 주위에 브랜드명을 출력
        - 힌트) geom_text(aes(label = brand), hjust = 0, vjust = 2)

실습 (Answer) : dplyr & magrittr

```
> # question 1
> mtcars %>%
+   select(mpg, cyl, disp, hp) %>%
+  mutate(brand = sapply(strsplit(row.names(.), split =" "),
+                                "[", 1)) %>%
+   filter(cyl >= 6) %>%
+   group_by(brand) %>%
+   summarise(max.disp = max(disp),
+             mean.mpg = mean(mpg))
```

실습 (Answer) : dplyr & magrittr

```
> # question 2
> mtcars %>%
+   select(mpg, cyl, disp, hp) %>%
+  mutate(brand = sapply(strsplit(row.names(.), split =" "),
+                                "[", 1)) %>%
+   filter(cyl >= 6) %>%
+   group_by(brand) %>%
+   summarise(max.disp = max(disp),
+             mean.mpg = mean(mpg)) %>%
+   ggplot(aes(x = max.disp, y = mean.mpg)) +
+   geom_point() +
+   geom_smooth(method = lm) +
+   geom_text(aes(label = brand), hjust = 0, vjust = 2)
```

Parallel Processing

Parallel Processing 관련 패키지

dplyr 패키지는 함수를 중심으로 데이터 조작을 수행함

표: 대표 패키지

Packages	UNIX	Windows	비고
parallel	O	O	parallel computation
doMC	O	X	foreach parallel adaptor
doParallel	O	O	foreach parallel adaptor
doSNOW	O	O	foreach parallel adaptor
multidplyr	O	O	parallel for dplyr

execrises: 준비1

%do% 연산자를 사용하는 사용자 정의함수 준비

```
> iris2 <- iris[which(iris[, 5] != "setosa"), c(1,5)]
> iris2[1:2, ]

   Sepal.Length    Species
51          7.0 versicolor
52          6.4 versicolor

> foreach1 <- function(x) {
+   foreach(icount(x), .combine = cbind) %do% {
+     ind <- sample(100, 100, replace = TRUE)
+     rst <- glm(iris2[ind, 2] ~ iris2[ind, 1],
+                family = binomial(logit))
+     coefficients(rst)}}
```

execrises: 준비2

%dopar% 연산자를 사용하는 사용자 정의함수 준비

```
> foreach2 <- function(x) {
+   foreach(icount(x), .combine = cbind) %dopar% {
+     ind <- sample(100, 100, replace = TRUE)
+     rst <- glm(iris2[ind, 2] ~ iris2[ind, 1],
+                family = binomial(logit))
+     coefficients(rst)}}
```

execrises: doMC

```
> library(doMC)
> registerDoMC(cores = 8)
> trials <- 10000
> system.time(r <- foreach1(trials))

   user  system elapsed
  21.58    0.83   22.42

> system.time(r <- foreach2(trials))

   user  system elapsed
 48.411   7.808   8.778
```

execrises: doSNOW

```
> library(doSNOW)
> cl <- makeCluster(8, type="SOCK")
> registerDoSNOW(cl)
> system.time({foreach(icount(trials),
+                      .combine = cbind) %dopar% {
+     ind <- sample(100, 100, replace = TRUE)
+     rst <- glm(iris2[ind, 2] ~ iris2[ind, 1],
+                family = binomial(logit))
+     coefficients(rst)}})

   user  system elapsed
  5.938   1.329   7.301

> stopCluster(cl)
```

execrises: doParallel

```
> library(doParallel)
> cl <- makePSOCKcluster(8)
> registerDoParallel(cl)
> system.time({foreach(icount(trials),
+                      .combine = cbind) %dopar% {
+     ind <- sample(100, 100, replace = TRUE)
+     rst <- glm(iris2[ind, 2] ~ iris2[ind, 1],
+                family = binomial(logit))
+     coefficients(rst)}})

   user  system elapsed
  4.694   1.296   6.290

> stopCluster(cl)
```

execrises: parallel 준비

parallel 패키지에서 병렬로 호출할 사용자 정의함수 준비

```
> mdl <- function(x) {
+   ind <- sample(100, 100, replace = TRUE)
+   rst <- glm(iris2[ind, 2] ~ iris2[ind, 1],
+              family = binomial(logit))
+     coefficients(rst)
+ }
```

execrises: parallel-UNIX

리스트로 변환된 결과를 sapply()로 행렬화함

```
> library(parallel)
> system.time(rst <- mclapply(X=seq(trials), FUN = mdl,
+                             mc.cores = 8))

   user  system elapsed
  0.029   0.078   5.473

> x <- sapply(rst, cbind)
```

execrises: parallel-UNIX/Windows

MS-Windows에서는 다음의 방법만 허용함

```
> library(parallel)
> cl <- makeCluster(8)
> system.time(rst <- mclapply(X=seq(trials), FUN = mdl))

   user  system elapsed
  0.029   0.067  10.437

> stopCluster(cl)
> x <- sapply(rst, cbind)
```

multidplyr package

multidplyr 패키지는 dplyr 패키지의 기능에 병렬처리를 지원

multidplyr parallel processing

1. partition()호출 : 데이터가 multiple cores로 분할
2. 개별 cores에서 작업 수행
3. collect() 호출 : 데이터 회수

execrises: multidplyr

parallel 기능을 사용하지 않은 방법, 그래도 느리지 않음

```
> library(dplyr)
> library(nycflights13)
> system.time({
+  flights %>%
+    group_by(flight) %>%
+    summarise(mean(dep_delay, na.rm = TRUE))})
   user  system elapsed
  0.290   0.028   0.319
```

execrises: multidplyr

parallel 기능을 사용했지만 더 느림(오버헤드) - 데이터가 작음

```
> library(multidplyr)
> dim(flights)
[1] 336776     19
> cluster <- new_cluster(8)
> system.time({
+  flights %>%
+    partition(cluster) %>%
+    partition(flight) %>%
+    summarise(mean(dep_delay, na.rm = TRUE)) %>%
+    collect()})
   user  system elapsed
  0.376   0.073   1.003
```

References

Getting Started with doMC and foreach

https://cran.r-project.org/web/packages/doMC/vignettes/gettingstartedMC.pdf

Steve Weston, 2015

An introduction to multidplyr

https://github.com/hadley/multidplyr/blob/master/vignettes/multidplyr.md

Hadley Wickham, 2015

Data Wrangling with dplyr and tidyr Cheat Sheet

https://www.rstudio.com/wp-content/uploads/2015/02/data-wrangling-cheatsheet.pdf

Rstudio, 2015

THE END