PR11

201821479\_황혜린

2018년 12월 1일

# 1 . dplyr 패키지

. dplyr은 Hadley Wickham이 만든 데이터 핸들링을 위한 패키지 . dplyr은 c++로 작성되어 기존 데이터핸들링 패키지보다 빠른 데이터조작이 가능 . 각종 데이터베어스 지원(MySQL, PostgreSQL, SQLite, BigQuery)수 . 기존R의 기본 문법과 프로그래밍능력을 바탕으로 데이터의 조작이 가능하지만 dplyr패키지를 활용하면 통일된 문법양식으로 데이터조작이 가능함 ·체인연산자를 지원함으로(%>%) 앞부분의 연산결과를 뒤에 오는 함수의 입력값으로 사용할 수 있음

## dplyr의 주요함수

filter; 지정한 조건식에 맞는 데이터 추출; subset select; 열의 추출; data[, c(“ Year”, “Month”)] mutate; 열 추가; transform arrange; 정렬; order, sort summarise; 집계; aggregate

# dplyr 실습 데이터 : nycflights13

. 미국 휴스턴에서 출발하는 모든 비행기의 이착륙기록

#install.packages("nycflights13")  
library(nycflights13)  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

head(flights)

## # A tibble: 6 x 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int>  
## 1 2013 1 1 517 515 2 830  
## 2 2013 1 1 533 529 4 850  
## 3 2013 1 1 542 540 2 923  
## 4 2013 1 1 544 545 -1 1004  
## 5 2013 1 1 554 600 -6 812  
## 6 2013 1 1 554 558 -4 740  
## # ... with 12 more variables: sched\_arr\_time <int>, arr\_delay <dbl>,  
## # carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,  
## # air\_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>,  
## # time\_hour <dttm>

# 1.1 dplyr의 filter함수

* 데이터에서 원하는 조건에 따라 행을 추출하는 함수
* filter( 데이터,조건1 | 조건2) : 조건1 또는 조건2 둘 중 한가지를 충족하는 데이터 추출
* filter( 데이터, 조건1 & 조건2) : 조건1과 조건2 모두 충족하는 데이터 추출 ·조건을 작성할 때 쉼표’,’는 AND, ‘|’는 OR와 같음

filter(flights, month ==1 | day==1)

## # A tibble: 37,198 x 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int>  
## 1 2013 1 1 517 515 2 830  
## 2 2013 1 1 533 529 4 850  
## 3 2013 1 1 542 540 2 923  
## 4 2013 1 1 544 545 -1 1004  
## 5 2013 1 1 554 600 -6 812  
## 6 2013 1 1 554 558 -4 740  
## 7 2013 1 1 555 600 -5 913  
## 8 2013 1 1 557 600 -3 709  
## 9 2013 1 1 557 600 -3 838  
## 10 2013 1 1 558 600 -2 753  
## # ... with 37,188 more rows, and 12 more variables: sched\_arr\_time <int>,  
## # arr\_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,  
## # origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>, hour <dbl>,  
## # minute <dbl>, time\_hour <dttm>

filter(flights, month ==1 , day==1)

## # A tibble: 842 x 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int>  
## 1 2013 1 1 517 515 2 830  
## 2 2013 1 1 533 529 4 850  
## 3 2013 1 1 542 540 2 923  
## 4 2013 1 1 544 545 -1 1004  
## 5 2013 1 1 554 600 -6 812  
## 6 2013 1 1 554 558 -4 740  
## 7 2013 1 1 555 600 -5 913  
## 8 2013 1 1 557 600 -3 709  
## 9 2013 1 1 557 600 -3 838  
## 10 2013 1 1 558 600 -2 753  
## # ... with 832 more rows, and 12 more variables: sched\_arr\_time <int>,  
## # arr\_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,  
## # origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>, hour <dbl>,  
## # minute <dbl>, time\_hour <dttm>

filter(flights, month ==1 , day==1, year==2013)

## # A tibble: 842 x 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int>  
## 1 2013 1 1 517 515 2 830  
## 2 2013 1 1 533 529 4 850  
## 3 2013 1 1 542 540 2 923  
## 4 2013 1 1 544 545 -1 1004  
## 5 2013 1 1 554 600 -6 812  
## 6 2013 1 1 554 558 -4 740  
## 7 2013 1 1 555 600 -5 913  
## 8 2013 1 1 557 600 -3 709  
## 9 2013 1 1 557 600 -3 838  
## 10 2013 1 1 558 600 -2 753  
## # ... with 832 more rows, and 12 more variables: sched\_arr\_time <int>,  
## # arr\_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,  
## # origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>, hour <dbl>,  
## # minute <dbl>, time\_hour <dttm>

# 1.2 dplyr의 arrange함수

. 데이터를 원하는 조건에 따라 정렬해주는 함수 \* arrange(데이터, 정렬기준컬럼1, 정렬기준컬럼2, 정렬기준컬럼3) . 내림차순으로 정렬 시 desc함수사용: arrange(데이터, desc(정렬기준컬럼1))

arrange(flights, year, month ,day)

## # A tibble: 336,776 x 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int>  
## 1 2013 1 1 517 515 2 830  
## 2 2013 1 1 533 529 4 850  
## 3 2013 1 1 542 540 2 923  
## 4 2013 1 1 544 545 -1 1004  
## 5 2013 1 1 554 600 -6 812  
## 6 2013 1 1 554 558 -4 740  
## 7 2013 1 1 555 600 -5 913  
## 8 2013 1 1 557 600 -3 709  
## 9 2013 1 1 557 600 -3 838  
## 10 2013 1 1 558 600 -2 753  
## # ... with 336,766 more rows, and 12 more variables: sched\_arr\_time <int>,  
## # arr\_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,  
## # origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>, hour <dbl>,  
## # minute <dbl>, time\_hour <dttm>

arrange(flights, desc(month))

## # A tibble: 336,776 x 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int>  
## 1 2013 12 1 13 2359 14 446  
## 2 2013 12 1 17 2359 18 443  
## 3 2013 12 1 453 500 -7 636  
## 4 2013 12 1 520 515 5 749  
## 5 2013 12 1 536 540 -4 845  
## 6 2013 12 1 540 550 -10 1005  
## 7 2013 12 1 541 545 -4 734  
## 8 2013 12 1 546 545 1 826  
## 9 2013 12 1 549 600 -11 648  
## 10 2013 12 1 550 600 -10 825  
## # ... with 336,766 more rows, and 12 more variables: sched\_arr\_time <int>,  
## # arr\_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,  
## # origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>, hour <dbl>,  
## # minute <dbl>, time\_hour <dttm>

# 1.3 dplyr의 select함수

* select함수는 원하는 열(column)을 추출
* select(데이터, 컬럼1, 컬럼2, 컬럼3)
* select(데이터, 컬럼1:컬럼3) -컬럼명을 변경할 수 있음

select(flights, year, month, day)

## # A tibble: 336,776 x 3  
## year month day  
## <int> <int> <int>  
## 1 2013 1 1  
## 2 2013 1 1  
## 3 2013 1 1  
## 4 2013 1 1  
## 5 2013 1 1  
## 6 2013 1 1  
## 7 2013 1 1  
## 8 2013 1 1  
## 9 2013 1 1  
## 10 2013 1 1  
## # ... with 336,766 more rows

select(flights, year:day)

## # A tibble: 336,776 x 3  
## year month day  
## <int> <int> <int>  
## 1 2013 1 1  
## 2 2013 1 1  
## 3 2013 1 1  
## 4 2013 1 1  
## 5 2013 1 1  
## 6 2013 1 1  
## 7 2013 1 1  
## 8 2013 1 1  
## 9 2013 1 1  
## 10 2013 1 1  
## # ... with 336,766 more rows

select(flights, -(year:day))

## # A tibble: 336,776 x 16  
## dep\_time sched\_dep\_time dep\_delay arr\_time sched\_arr\_time arr\_delay  
## <int> <int> <dbl> <int> <int> <dbl>  
## 1 517 515 2 830 819 11  
## 2 533 529 4 850 830 20  
## 3 542 540 2 923 850 33  
## 4 544 545 -1 1004 1022 -18  
## 5 554 600 -6 812 837 -25  
## 6 554 558 -4 740 728 12  
## 7 555 600 -5 913 854 19  
## 8 557 600 -3 709 723 -14  
## 9 557 600 -3 838 846 -8  
## 10 558 600 -2 753 745 8  
## # ... with 336,766 more rows, and 10 more variables: carrier <chr>,  
## # flight <int>, tailnum <chr>, origin <chr>, dest <chr>, air\_time <dbl>,  
## # distance <dbl>, hour <dbl>, minute <dbl>, time\_hour <dttm>

# 1.4 dplyr의 distinct함수

* 중복항목을 제외한 데이터를 확인 할 수 있음(unique함수와 동일) -distinct(데이터, 컬럼명)

distinct(select(flights, tailnum))

## # A tibble: 4,044 x 1  
## tailnum  
## <chr>   
## 1 N14228   
## 2 N24211   
## 3 N619AA   
## 4 N804JB   
## 5 N668DN   
## 6 N39463   
## 7 N516JB   
## 8 N829AS   
## 9 N593JB   
## 10 N3ALAA   
## # ... with 4,034 more rows

distinct(select(flights, origin, dest))

## # A tibble: 224 x 2  
## origin dest   
## <chr> <chr>  
## 1 EWR IAH   
## 2 LGA IAH   
## 3 JFK MIA   
## 4 JFK BQN   
## 5 LGA ATL   
## 6 EWR ORD   
## 7 EWR FLL   
## 8 LGA IAD   
## 9 JFK MCO   
## 10 LGA ORD   
## # ... with 214 more rows

# 1.5 dplyr의 mutate함수

-기존 데이티 프레임에 새로운 열을 추가해줌 \* 데이터 프레임 내의 변수들을 활용해 새로운 변수를 만들 때 효과적임 ·새로 생성한 변수를 해당 함수내에서 바로 활용이 가능

mutate(flights, gain <- arr\_delay - dep\_delay)

## # A tibble: 336,776 x 20  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int>  
## 1 2013 1 1 517 515 2 830  
## 2 2013 1 1 533 529 4 850  
## 3 2013 1 1 542 540 2 923  
## 4 2013 1 1 544 545 -1 1004  
## 5 2013 1 1 554 600 -6 812  
## 6 2013 1 1 554 558 -4 740  
## 7 2013 1 1 555 600 -5 913  
## 8 2013 1 1 557 600 -3 709  
## 9 2013 1 1 557 600 -3 838  
## 10 2013 1 1 558 600 -2 753  
## # ... with 336,766 more rows, and 13 more variables: sched\_arr\_time <int>,  
## # arr\_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,  
## # origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>, hour <dbl>,  
## # minute <dbl>, time\_hour <dttm>, `gain <- arr\_delay - dep\_delay` <dbl>

mutate(flights, gain=arr\_delay - dep\_delay, gain\_per\_hour=gain/(air\_time/60))

## # A tibble: 336,776 x 21  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int>  
## 1 2013 1 1 517 515 2 830  
## 2 2013 1 1 533 529 4 850  
## 3 2013 1 1 542 540 2 923  
## 4 2013 1 1 544 545 -1 1004  
## 5 2013 1 1 554 600 -6 812  
## 6 2013 1 1 554 558 -4 740  
## 7 2013 1 1 555 600 -5 913  
## 8 2013 1 1 557 600 -3 709  
## 9 2013 1 1 557 600 -3 838  
## 10 2013 1 1 558 600 -2 753  
## # ... with 336,766 more rows, and 14 more variables: sched\_arr\_time <int>,  
## # arr\_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,  
## # origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>, hour <dbl>,  
## # minute <dbl>, time\_hour <dttm>, gain <dbl>, gain\_per\_hour <dbl>

# 1.6 dplyr의 summarise함수

-mean(), sd(), var(), median()함수를 활용해 기술통계량을 확인 -결과를 데이터프레임으로 반환함

summarise(flights, delay = mean(dep\_delay, na.rm = T))

## # A tibble: 1 x 1  
## delay  
## <dbl>  
## 1 12.6

# 1.7 dplyr의 group\_by()함수

* 변수의 레벨에 따라 자료를 그룹화 해줌 \*그룹에 따른 수치자료를 산출하고 싶을 때 편리함
* summarise함수와 함께 사용시 aggregate함수와 같은 기능
* ex)직급에 따른 평균 연봉과 사용가능한 연차일수(휴가)를 구하고 싶을 때!

#비행기별로 그룹 만들기  
  
by\_tailnum = group\_by(flights, tailnum)  
  
delay = summarise(by\_tailnum, count=n(), dist=mean(distance, na.rm = T), delay=mean(arr\_delay, na.rm = T))  
  
   
#횟수가 20이상, 거리가 2000이하인 비행기만 추출  
  
delay=filter(delay, count > 20, dist < 2000)  
dist

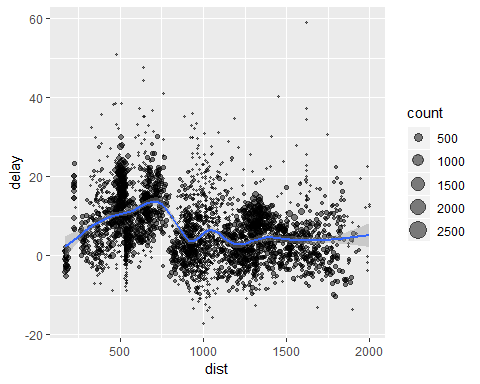
## function (x, method = "euclidean", diag = FALSE, upper = FALSE,   
## p = 2)   
## {  
## if (!is.na(pmatch(method, "euclidian")))   
## method <- "euclidean"  
## METHODS <- c("euclidean", "maximum", "manhattan", "canberra",   
## "binary", "minkowski")  
## method <- pmatch(method, METHODS)  
## if (is.na(method))   
## stop("invalid distance method")  
## if (method == -1)   
## stop("ambiguous distance method")  
## x <- as.matrix(x)  
## N <- nrow(x)  
## attrs <- if (method == 6L)   
## list(Size = N, Labels = dimnames(x)[[1L]], Diag = diag,   
## Upper = upper, method = METHODS[method], p = p, call = match.call(),   
## class = "dist")  
## else list(Size = N, Labels = dimnames(x)[[1L]], Diag = diag,   
## Upper = upper, method = METHODS[method], call = match.call(),   
## class = "dist")  
## .Call(C\_Cdist, x, method, attrs, p)  
## }  
## <bytecode: 0x000000001be6b058>  
## <environment: namespace:stats>

#위에서 만든 delay데이터로 시각화  
  
library(ggplot2)  
  
ggplot(delay, aes(dist, delay)) +   
 geom\_point(aes(size = count), alpha = 1/2) +  
 geom\_smooth() +   
 scale\_size\_area()

## `geom\_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'

## Warning: Removed 1 rows containing non-finite values (stat\_smooth).

## Warning: Removed 1 rows containing missing values (geom\_point).



# 1.8 dplyr의 Chain 연산

-여러단계의 함수나 연산을 연결하여 한번에 수행할 때 사용 \* 옆의 함수의 결과는 바로 뒤에 오는 함수의 입력값이 됨 \* 데이터를 여러 객체에 할당하지 않아도 되기때문에 메모리 관리에 유리함

## 체인연산 사용하지 않을 때

a1= group\_by(flights, year, month, day)  
a2= select(a1, year:day, arr\_delay)  
a3= summarise(a2, arr=mean(arr\_delay, na.rm=T))  
a4= filter(a3, arr >30)  
a4

## # A tibble: 42 x 4  
## # Groups: year, month [11]  
## year month day arr  
## <int> <int> <int> <dbl>  
## 1 2013 1 16 34.2  
## 2 2013 1 31 32.6  
## 3 2013 2 11 36.3  
## 4 2013 2 27 31.3  
## 5 2013 3 8 85.9  
## 6 2013 3 18 41.3  
## 7 2013 4 10 38.4  
## 8 2013 4 12 36.0  
## 9 2013 4 18 36.0  
## 10 2013 4 19 47.9  
## # ... with 32 more rows

## 체인연산 사용했을 때

flights %>% group\_by(year, month, day) %>% select(arr\_delay) %>% summarise(arr = mean(arr\_delay, na.rm=T)) %>% filter(arr>30)

## Adding missing grouping variables: `year`, `month`, `day`

## # A tibble: 42 x 4  
## # Groups: year, month [11]  
## year month day arr  
## <int> <int> <int> <dbl>  
## 1 2013 1 16 34.2  
## 2 2013 1 31 32.6  
## 3 2013 2 11 36.3  
## 4 2013 2 27 31.3  
## 5 2013 3 8 85.9  
## 6 2013 3 18 41.3  
## 7 2013 4 10 38.4  
## 8 2013 4 12 36.0  
## 9 2013 4 18 36.0  
## 10 2013 4 19 47.9  
## # ... with 32 more rows

# 1.9 dplyr의 join함수

-join(x, y) 또는 join(x, y, by=”기준열”) 형태 -조인의 기준이 되는 단일 컬럼이 존재하는 경우 별도 by인수를 지정하지 않아도 됨

#join 실습 데이터 생성  
#install.packages("readr")  
  
library(readr)  
superheroes <- "  
name, alignment, gender, publisher  
Magneto, bad, male, Marvel  
Storm, good, female, Marvel  
Mystique, bad, female, Marvel  
Betman, good, male, DC  
Joker, bad, male, DC  
Catwoman, bad, female, DC  
Heilboy, good, male, Dark Horse Comics  
"  
  
publishers <- "  
Publisher, yr\_founded  
DC, 1934  
Marvel, 1939  
Image, 1992  
"  
superheroes <- read\_csv(superheroes, trim\_ws=T)  
publishers <- read\_csv(publishers, trim\_ws=T)

### inner\_join, Ieft\_join, full\_join, anti\_join, semi\_join 각각의 출력 값 확인 하기

library(dplyr)  
#inner\_join(superheroes, publishers)  
#left\_join(superheroes, publishers)  
#full\_join(superheroes, publishers)  
#anti\_join(superheroes, publishers)  
#semi\_join(superheroes, publishers)  
  
#ㅠㅠ계속 에러가 떠서 주석처리합니다

# join 예제

-raw.csv는 국회의원 별 의원비 지출내역 -join.csv는 국회의원 명단 -두 데이터를 조인하기 위해서는 raw.csv의 데이터를 -aggregate 또는 summarize함수를 사용해 요약해야 함 \* 이름을 기준으로 join을 할 것

#의원비 지출이 가장 많은 10명의 직업과 학력을 확인   
#데이터 로딩  
  
setwd("C:/Users/hyere/Desktop/R/CSVdata")  
raw\_data=read.csv("raw.csv")  
join\_data = read.csv("join.csv")  
head(raw\_data)

## X total\_num cong\_num name party region date  
## 1 1 217616 309 현영희 무소속 비례 2014-01-03  
## 2 2 217617 309 현영희 무소속 비례 2014-01-03  
## 3 3 217618 309 현영희 무소속 비례 2014-01-03  
## 4 4 217619 309 현영희 무소속 비례 2014-01-03  
## 5 5 217620 309 현영희 무소속 비례 2014-01-05  
## 6 6 221930 1 강기윤 새누리당 경남 창원성산구 2014-01-02  
## item expense store category region2  
## 1 항공료 68600 대한항공 교통-항공 비례  
## 2 전문가 간담회 식대 78000 카페모차르트 간담회-식대 비례  
## 3 항공료 97100 대한항공 교통-항공 비례  
## 4 유류비 138000 예스제이오일 차량-주유 비례  
## 5 유류비 60000 과정로셀프주유소 차량-주유 비례  
## 6 우편발송료 3070 창원우체국 홍보-우편 경남  
## date2 date\_month  
## 1 2014-01-03 1월  
## 2 2014-01-03 1월  
## 3 2014-01-03 1월  
## 4 2014-01-03 1월  
## 5 2014-01-05 1월  
## 6 2014-01-02 1월

head(join\_data)

## X region district party name gender birth  
## 1 94 비례 비례대표 새누리당 민현주 여 1969-07-23  
## 2 135 비례 비례대표 새누리당 신경림 여 1954-03-22  
## 3 235 경기 하남시 새누리당 이현재 남 1949-04-25  
## 4 294 인천 남구갑 새누리당 홍일표 남 1956-02-11  
## 5 185 충북 충주시 새누리당 윤진식 남 1946-03-04  
## 6 131 비례 비례대표 새누리당 손인춘 여 1959-05-13  
## job  
## 1 경기대학교 일반대학원 직업학과 조교수  
## 2 이화여자대학교 건강과학대학학장  
## 3 정당인  
## 4 국회의원  
## 5 국회의원  
## 6 (주)인성내츄럴 사장  
## achievement  
## 1 Cornell 대학교 대학원 사회학과졸업(박사), (1999.8~2004.8)  
## 2 Teachers College, Columbia University 대학원졸업(교육학박사), (1989.9~1992.2)  
## 3 건국대학교 대학원 경영학박사  
## 4 건국대학교 대학원 법학과 졸업(법학석사)  
## 5 건국대학교 일반대학원 국제무역학과 졸업(경제학박사)  
## 6 건국대학교 일반대학원 박사과정 벤처전문기술학과 벤처기술경영전공수료  
## career vote\_rate age regeion job2  
## 1 (현)경기대학교 일반대학원 직업학과 조교수 48 비례 교수  
## 2 (현)한국여성단체협의회 수석부회장 63 비례 교수  
## 3 (전)중소기업청장 33,861 68 경기 정당인  
## 4 (전)인천광역시 정무부시장 40,195 61 인천 국회의원  
## 5 전)산업자원부장관 56,688 71 충북 국회의원  
## 6 (현)(주)인성내츄럴 사장 58 비례 기업인  
## achievement.1 X.1 X.2 X.3 X.4  
## 1 Cornell NA  
## 2 Columbia NA  
## 3 건국대학교 NA  
## 4 건국대학교 NA  
## 5 건국대학교 NA  
## 6 건국대학교 NA

#조인할 데이터 추출  
expense=aggregate(expense~name+party, data=raw\_data, FUN=sum)  
job\_school=select(join\_data, name, job, achievement)  
#조인   
joined =inner\_join(expense, job\_school, by='name')

## Warning: Column `name` joining factors with different levels, coercing to  
## character vector

joined = joined[order(joined$expense, decreasing=T),]  
joined[1:10,]

## name party expense job  
## 91 이명수 새누리당 446913569 국회의원  
## 17 김상민 새누리당 402282444 대학생자원봉사단 V원정대 대표  
## 126 조원진 새누리당 392767643 국회의원  
## 58 서상기 새누리당 351375323 국회의원  
## 277 이상규 통합진보 346551904 정당인  
## 271 심상정 진보정의 336436592 정당인  
## 187 박지원 새정치 331136715 국회의원  
## 120 정병국 새누리당 330634706 국회의원  
## 215 유성엽 새정치 330615390 국회의원  
## 233 이종걸 새정치 323533338 변호사  
## achievement  
## 91 성균관대 대학원 졸업(행정학과 행정학 박사)  
## 17 아주대학교 사학과졸업  
## 126 영남대학교 행정대학원 정책분석학과 졸업(행정학 석사)  
## 58 美드렉셀대학교 공학박사(1972.09~1976.06)  
## 277 서울대학교 법과대학 공법학과 졸업  
## 271 서울대학교 사회교육과 졸업  
## 187 단국대 상학과 졸업  
## 120 성균관 대학교 대학원 졸업(정치학박사)  
## 215 서울대학교 사회과학대학 외교학과 졸업  
## 233 서울대학교 법과대학 공법학과 졸업

# 2. reshape2

* reshape2는 wide와 long포맷으로 쉽게 데이터를 변환해줌 . melt 함수는 wide 포맷 데이터를 long 포맷으로 변환해주는 함수 (tidyr::gather와 유사)
* cast는 long 포맷 데이터를 wide 포맷으로 변환해주는 함수 (tidyr::spread와 유사)

# 2.1. wide 포맷 데이터를 long 포맷으로 전환하기: melt 함수

* R에 내장되어 있는 airquality라는 데이터셋 사용
* 먼저 데이터셋을 쉽게 변경하기 위해 각 열의 이름을 전부 소문자로 변경

library(reshape2)  
names(airquality) <- tolower(names(airquality))  
head(airquality)

## 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  
## 4 18 313 11.5 62 5 4  
## 5 NA NA 14.3 56 5 5  
## 6 28 NA 14.9 66 5 6

#일단위 오존량, 태양 복사열, 풍속 그리고 온도를 알고 싶습니다.  
#이럴 때 “ID variable”을 사용하면 설정을 해 줄 수 있습니다.  
  
aql <- melt(airquality, id.vars=c("month", "day"))  
head(aql)

## month day variable value  
## 1 5 1 ozone 41  
## 2 5 2 ozone 36  
## 3 5 3 ozone 12  
## 4 5 4 ozone 18  
## 5 5 5 ozone NA  
## 6 5 6 ozone 28

#long 포맷 데이터의 열 이름 지정  
aql <- melt(airquality, id.vars=c("month", "day"), variable.names=("climate\_variable"), value.name = ("climate\_value"))  
head(aql)

## month day variable climate\_value  
## 1 5 1 ozone 41  
## 2 5 2 ozone 36  
## 3 5 3 ozone 12  
## 4 5 4 ozone 18  
## 5 5 5 ozone NA  
## 6 5 6 ozone 28

## melt 연습하기

#Chickweight데이터(R 기본 내장 데이터셋)  
  
head(ChickWeight)

## Grouped Data: weight ~ Time | Chick  
## weight Time Chick Diet  
## 1 42 0 1 1  
## 2 51 2 1 1  
## 3 59 4 1 1  
## 4 64 6 1 1  
## 5 76 8 1 1  
## 6 93 10 1 1

#melt를 사용하여 ID variable을 time,chick,diet를 주고 weigh를value로 주는 chick-m이란 변수 만들기