PR12-Data Wrangling

조성우

2020 6월 10일

# 1.Data Wrangling with tidyverse

* Data wrangling이란, 분석을 진행하기위해 날것(raw)의 데이터를 분석에 적합한 형태로 정형화시키는 작업입니다,
* R에서는 tidyverse라는 패키지 생태계를 구성하고 있어서, 일관성있고 쉬운 작업을 가능하게 합니다.

# install.packages("tidyverse")  
library(tidyverse)

## Warning: package 'tidyverse' was built under R version 3.6.3

## -- Attaching packages -------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------- tidyverse 1.3.0 --

## √ ggplot2 3.3.0.9000 √ purrr 0.3.3   
## √ tibble 2.1.3 √ dplyr 0.8.5   
## √ tidyr 1.0.2 √ stringr 1.4.0   
## √ readr 1.3.1 √ forcats 0.5.0

## Warning: package 'tidyr' was built under R version 3.6.3

## Warning: package 'readr' was built under R version 3.6.3

## Warning: package 'dplyr' was built under R version 3.6.3

## Warning: package 'stringr' was built under R version 3.6.3

## Warning: package 'forcats' was built under R version 3.6.3

## -- Conflicts ----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

# 2.tidyr

\*tidyr은 Handley wicham이 만든 데이터의 포맷을 변경하기위한 패키지

## tidyr의 주요함수

#gather() : 데이터를 wide에서 long포맷으로 변경  
#spread() : 데이터를 long에서 wide 포맷으로 변겯  
#seperate() : 단일 열(column)을 복수 열들로 분리  
#unite() : 복수 열(column)들을 단일 열로 결합

# tidyr 실습데이터 : cases in EDAWR

\*Dataset to support the Expert Data Analysis with R : EDAWR

#install.packages("devtools")  
#devtools::install\_github("rstudio/EDAWR")  
  
library(EDAWR)

##   
## Attaching package: 'EDAWR'

## The following object is masked from 'package:dplyr':  
##   
## storms

## The following objects are masked from 'package:tidyr':  
##   
## population, who

head(cases)

## country 2011 2012 2013  
## 1 FR 7000 6900 7000  
## 2 DE 5800 6000 6200  
## 3 US 15000 14000 13000

head(pollution)

## city size amount  
## 1 New York large 23  
## 2 New York small 14  
## 3 London large 22  
## 4 London small 16  
## 5 Beijing large 121  
## 6 Beijing small 56

head(storms)

## storm wind pressure date  
## 1 Alberto 110 1007 2000-08-03  
## 2 Alex 45 1009 1998-07-27  
## 3 Allison 65 1005 1995-06-03  
## 4 Ana 40 1013 1997-06-30  
## 5 Arlene 50 1010 1999-06-11  
## 6 Arthur 45 1010 1996-06-17

## 2.1.gather() 함수

* wide포맷의 데이터를 원하는 조건에 맞게 long포맷으로 변환하는 함수
* gather(데이터,키(key),값(Value),…) \*…: 원데이터로 부터 모으기(gather)가 진행될 열들의 범위

gather(cases, Year,n,2:4)

## country Year n  
## 1 FR 2011 7000  
## 2 DE 2011 5800  
## 3 US 2011 15000  
## 4 FR 2012 6900  
## 5 DE 2012 6000  
## 6 US 2012 14000  
## 7 FR 2013 7000  
## 8 DE 2013 6200  
## 9 US 2013 13000

## 2.2.spread() 함수

* long포맷의 데이터를 원하는 조건에 맞게 long포맷으로 변환하는 함수
* seperate(데이터,키(key),값(value),~) \*\* 키(key):복수개의 열로 spread될 기존 long포맷의 열이름 \*\*값(Value):복수개의 열로 spread되어 값이 될 기존 long포맷의 열이름

spread(pollution,size,amount)

## city large small  
## 1 Beijing 121 56  
## 2 London 22 16  
## 3 New York 23 14

## 2.3. separate()함수

* 하나의 열을 특정 조건에 따라 여러개의 열로 나누어주는 함수 입니다.
* separate(data,col,into,sep,~) \*\* col:조건에 따른 분할을 진행할 열이름 \*\* into:분할된 결과가 저장될 각 열들의 이름 \*\* sep:분할 조건

storms2 <- separate(storms, date,c("year","month","day"),sep="-")  
storms2

## # A tibble: 6 x 6  
## storm wind pressure year month day   
## <chr> <int> <int> <chr> <chr> <chr>  
## 1 Alberto 110 1007 2000 08 03   
## 2 Alex 45 1009 1998 07 27   
## 3 Allison 65 1005 1995 06 03   
## 4 Ana 40 1013 1997 06 30   
## 5 Arlene 50 1010 1999 06 11   
## 6 Arthur 45 1010 1996 06 17

## 2.4.unite() 함수

* 여러개로 나누어진 열을 특정 조건에 따라 결합해주는 함수입니다.
* unite(data,col,…,sep) \*\* col:조건에 따라 결합된 결과가 저장될 열이름 **…:합쳐질 열이름들**  sep:결합시 구분자

unite(storms2,"date",year,month,day,sep="-")

## # A tibble: 6 x 4  
## storm wind pressure date   
## <chr> <int> <int> <chr>   
## 1 Alberto 110 1007 2000-08-03  
## 2 Alex 45 1009 1998-07-27  
## 3 Allison 65 1005 1995-06-03  
## 4 Ana 40 1013 1997-06-30  
## 5 Arlene 50 1010 1999-06-11  
## 6 Arthur 45 1010 1996-06-17

# 3. dplyr

* dplyr은 Hadley Wickham이 만든 데이터 핸들링을 위한 패키지
* dplyr은 C++로 작성되어 기존 데이터핸들링 패키지보다 빠른조작이 가능
* 각종 데이터베이스 지원(MYSQL,Postgresql,SQLite,BigQeury)
* R의 기본문법과 프로그래밍능력만으로도 데이터의 조작이 가능하지만,dplyr 패키지를 활용하면 통일된 문법양식으로 데이터조작이 가능함
* 체인연산자를 지원함으로(%>%) 앞부분의 연산결과를 뒤에 오는 함수의 입력값으로 사용할 수 있음

### dplyr의 주요함수

#filter() : 지정한 조건식에 맞는 데이터를 추출  
#arrange() : 정렬  
#select() : 열의 추출  
#mutate() : 열추가  
#summarise() ; 집계

### dplyr 실습데이터 :nycflights13

\*미국 휴스턴에서 출발하는 모든 비행기의 이착륙

#install.packages("nycflights13) #해당패키지에 데이터가 있음  
library(nycflights13)

## Warning: package 'nycflights13' was built under R version 3.6.3

library(dplyr)  
head(flights)

## # A tibble: 6 x 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time sched\_arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int> <int>  
## 1 2013 1 1 517 515 2 830 819  
## 2 2013 1 1 533 529 4 850 830  
## 3 2013 1 1 542 540 2 923 850  
## 4 2013 1 1 544 545 -1 1004 1022  
## 5 2013 1 1 554 600 -6 812 837  
## 6 2013 1 1 554 558 -4 740 728  
## # ... with 11 more variables: 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>

### 3.1 filter() 함수

* 데이터에서 원하는 조건에 따라 행을 추출하는 함수
* filter(데이터,조건1ㅣ조건2):조건1 또는 조건2 둘중 한가지를 충족하는 데이터를 추출 \*조건을 작성할때는 쉼표’,’는 AND,’ㅣ’는 OR와 같음

filter(flights,month ==1 | day ==1)#37198row

## # A tibble: 37,198 x 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time sched\_arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int> <int>  
## 1 2013 1 1 517 515 2 830 819  
## 2 2013 1 1 533 529 4 850 830  
## 3 2013 1 1 542 540 2 923 850  
## 4 2013 1 1 544 545 -1 1004 1022  
## 5 2013 1 1 554 600 -6 812 837  
## 6 2013 1 1 554 558 -4 740 728  
## 7 2013 1 1 555 600 -5 913 854  
## 8 2013 1 1 557 600 -3 709 723  
## 9 2013 1 1 557 600 -3 838 846  
## 10 2013 1 1 558 600 -2 753 745  
## # ... with 37,188 more rows, and 11 more variables: 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) #842row

## # A tibble: 37,198 x 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time sched\_arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int> <int>  
## 1 2013 1 1 517 515 2 830 819  
## 2 2013 1 1 533 529 4 850 830  
## 3 2013 1 1 542 540 2 923 850  
## 4 2013 1 1 544 545 -1 1004 1022  
## 5 2013 1 1 554 600 -6 812 837  
## 6 2013 1 1 554 558 -4 740 728  
## 7 2013 1 1 555 600 -5 913 854  
## 8 2013 1 1 557 600 -3 709 723  
## 9 2013 1 1 557 600 -3 838 846  
## 10 2013 1 1 558 600 -2 753 745  
## # ... with 37,188 more rows, and 11 more variables: 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)#832row

## # A tibble: 842 x 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time sched\_arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int> <int>  
## 1 2013 1 1 517 515 2 830 819  
## 2 2013 1 1 533 529 4 850 830  
## 3 2013 1 1 542 540 2 923 850  
## 4 2013 1 1 544 545 -1 1004 1022  
## 5 2013 1 1 554 600 -6 812 837  
## 6 2013 1 1 554 558 -4 740 728  
## 7 2013 1 1 555 600 -5 913 854  
## 8 2013 1 1 557 600 -3 709 723  
## 9 2013 1 1 557 600 -3 838 846  
## 10 2013 1 1 558 600 -2 753 745  
## # ... with 832 more rows, and 11 more variables: 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>

## 3.2 arrange() 함수

* 데이터를 원한느 조건에 따라 정렬해주는 함수
* arrange(데이터,정렬기준컬럼1,정렬기준컬럼2,정렬기준컬럼3)
* 내림차순으로 정렬시 desc함수 사용: arrange(데이터,desc(정렬기준컬럼1))

arrange(flights,year,month,day)# ArrDelay, Month,Year 순으로 정렬

## # A tibble: 336,776 x 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time sched\_arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int> <int>  
## 1 2013 1 1 517 515 2 830 819  
## 2 2013 1 1 533 529 4 850 830  
## 3 2013 1 1 542 540 2 923 850  
## 4 2013 1 1 544 545 -1 1004 1022  
## 5 2013 1 1 554 600 -6 812 837  
## 6 2013 1 1 554 558 -4 740 728  
## 7 2013 1 1 555 600 -5 913 854  
## 8 2013 1 1 557 600 -3 709 723  
## 9 2013 1 1 557 600 -3 838 846  
## 10 2013 1 1 558 600 -2 753 745  
## # ... with 336,766 more rows, and 11 more variables: 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)) #Month 컬럼기준으로 내림차순으로 정렬

## # A tibble: 336,776 x 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time sched\_arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int> <int>  
## 1 2013 12 1 13 2359 14 446 445  
## 2 2013 12 1 17 2359 18 443 437  
## 3 2013 12 1 453 500 -7 636 651  
## 4 2013 12 1 520 515 5 749 808  
## 5 2013 12 1 536 540 -4 845 850  
## 6 2013 12 1 540 550 -10 1005 1027  
## 7 2013 12 1 541 545 -4 734 755  
## 8 2013 12 1 546 545 1 826 835  
## 9 2013 12 1 549 600 -11 648 659  
## 10 2013 12 1 550 600 -10 825 854  
## # ... with 336,766 more rows, and 11 more variables: 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>

## 3.3. select() 함수

* select함수는 원하는 열(column)을 추출
* select(데이터,컬럼1,컬럼2,컬럼3)
* select(데이터,컬럼1:컬럼3)
* 컬럼명을 변경할수 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 carrier  
## <int> <int> <dbl> <int> <int> <dbl> <chr>   
## 1 517 515 2 830 819 11 UA   
## 2 533 529 4 850 830 20 UA   
## 3 542 540 2 923 850 33 AA   
## 4 544 545 -1 1004 1022 -18 B6   
## 5 554 600 -6 812 837 -25 DL   
## 6 554 558 -4 740 728 12 UA   
## 7 555 600 -5 913 854 19 B6   
## 8 557 600 -3 709 723 -14 EV   
## 9 557 600 -3 838 846 -8 B6   
## 10 558 600 -2 753 745 8 AA   
## # ... with 336,766 more rows, and 9 more variables: flight <int>,  
## # tailnum <chr>, origin <chr>, dest <chr>, air\_time <dbl>, distance <dbl>,  
## # hour <dbl>, minute <dbl>, time\_hour <dttm>

## 3.4 distinct() 함수

* 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

## 3.5. mutate() 함수

* 기존 데이터 프레임에 새로운 열을 추가해줌
* 데이터프레임 내의 변수들을 활용해 새로운 변수를 만들때 효과적임
* 새로 생성한 변수를 해당 함수 내에서 바로 활용이 가능

#arr\_delay - dep\_delay값으로 gain컬럼 추가  
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 sched\_arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int> <int>  
## 1 2013 1 1 517 515 2 830 819  
## 2 2013 1 1 533 529 4 850 830  
## 3 2013 1 1 542 540 2 923 850  
## 4 2013 1 1 544 545 -1 1004 1022  
## 5 2013 1 1 554 600 -6 812 837  
## 6 2013 1 1 554 558 -4 740 728  
## 7 2013 1 1 555 600 -5 913 854  
## 8 2013 1 1 557 600 -3 709 723  
## 9 2013 1 1 557 600 -3 838 846  
## 10 2013 1 1 558 600 -2 753 745  
## # ... with 336,766 more rows, and 12 more variables: 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>

#gain컬럼을 만드는 동시에 gain컬럼을 이용해 다수의 다른 변수를 생성가능  
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 sched\_arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int> <int>  
## 1 2013 1 1 517 515 2 830 819  
## 2 2013 1 1 533 529 4 850 830  
## 3 2013 1 1 542 540 2 923 850  
## 4 2013 1 1 544 545 -1 1004 1022  
## 5 2013 1 1 554 600 -6 812 837  
## 6 2013 1 1 554 558 -4 740 728  
## 7 2013 1 1 555 600 -5 913 854  
## 8 2013 1 1 557 600 -3 709 723  
## 9 2013 1 1 557 600 -3 838 846  
## 10 2013 1 1 558 600 -2 753 745  
## # ... with 336,766 more rows, and 13 more variables: 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>

## 3.6 summarise() 함수

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

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

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

## 3.7 group\_by() 함수

* 변수의 레벨에 따라 자료를 그룹화해줌
* 그룹에 따른 수치자료를 산출하고 싶을때 편리함
* summarize함수와 함께 사용시 aggregate함수와 같은기능
* ex)직급에 따른 평균 연봉과 사용가능한 연차일수(휴가)를 구하고 싶을때!

#비행기별로 그룹만들기  
by\_tailnum <- group\_by(flights, tailnum) #비행기별로 그룹만들기  
  
#비행기별 비행회수,비행거리평균,연착시간평균 산출  
delay <- summarise(by\_tailnum,count=n(),dist=mean(distance,na.rm=TRUE),  
 delay = mean(arr\_delay,na.rm=TRUE))  
  
#회수가 20회 이상 거리가 2000이하인 비행기만 추출  
delay <-filter(delay,count>20,dist<2000)

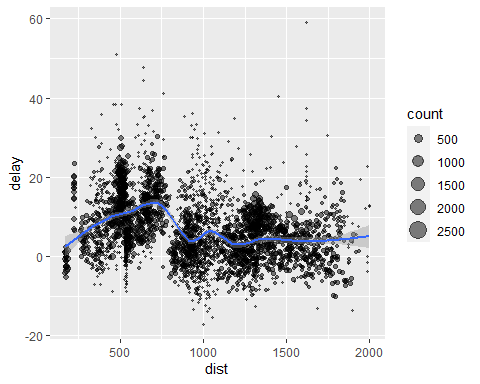
\*위에서 만든 delay데이터로 시각화

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).



## 3.8. join()함수

* join(x,y)또는 join(x,y,by=“기준열”)형태
* 조인의 기준이 되는 단일컬럼이 존재하는 경우 별도 by인수를 저장하지 않아도 됨
* 조인의 기준이 되는 컬럼이 여러개이거나, 여러가지 컬럼을 동시에 활용해야하는경우 by인수를 사용

#join 실습 데이터 생성  
  
superheroes <- "  
name, alignment,gender, publisher  
Magneto, bad, male, Marvel  
Storm, good, female, Marvel  
Mystique, bad ,female ,Marvel  
Batman, good , male ,DC  
Joker , bad, male, DC  
Catwomer, bad ,female, DC  
Hellboy, good male, Dark Horese Comics  
"  
  
publishers <- "  
publisher, yr\_founded  
DC, 1934  
Marvel, 1939  
Image, 1992  
"  
  
superheroes <- read\_csv(superheroes,trim\_ws = TRUE, skip = 1)

## Warning: 1 parsing failure.  
## row col expected actual file  
## 7 -- 4 columns 3 columns literal data

publishers <-read\_csv(publishers,trim\_ws= TRUE,skip=1)

* inner\_join, left\_join,anti\_join,semi\_join 각각의 출력값 확인하기

inner\_join(superheroes,publishers) #X,Y의 교집합

## Joining, by = "publisher"

## # A tibble: 6 x 5  
## name alignment gender publisher yr\_founded  
## <chr> <chr> <chr> <chr> <dbl>  
## 1 Magneto bad male Marvel 1939  
## 2 Storm good female Marvel 1939  
## 3 Mystique bad female Marvel 1939  
## 4 Batman good male DC 1934  
## 5 Joker bad male DC 1934  
## 6 Catwomer bad female DC 1934

left\_join(superheroes,publishers) #X기준(왼쪽)으로 머징

## Joining, by = "publisher"

## # A tibble: 7 x 5  
## name alignment gender publisher yr\_founded  
## <chr> <chr> <chr> <chr> <dbl>  
## 1 Magneto bad male Marvel 1939  
## 2 Storm good female Marvel 1939  
## 3 Mystique bad female Marvel 1939  
## 4 Batman good male DC 1934  
## 5 Joker bad male DC 1934  
## 6 Catwomer bad female DC 1934  
## 7 Hellboy good male Dark Horese Comics <NA> NA

full\_join(superheroes,publishers)# X,Y의 합집합

## Joining, by = "publisher"

## # A tibble: 8 x 5  
## name alignment gender publisher yr\_founded  
## <chr> <chr> <chr> <chr> <dbl>  
## 1 Magneto bad male Marvel 1939  
## 2 Storm good female Marvel 1939  
## 3 Mystique bad female Marvel 1939  
## 4 Batman good male DC 1934  
## 5 Joker bad male DC 1934  
## 6 Catwomer bad female DC 1934  
## 7 Hellboy good male Dark Horese Comics <NA> NA  
## 8 <NA> <NA> <NA> Image 1992

anti\_join(superheroes,publishers) #X의 컬럼만 유지하여 머징

## Joining, by = "publisher"

## # A tibble: 1 x 4  
## name alignment gender publisher  
## <chr> <chr> <chr> <chr>   
## 1 Hellboy good male Dark Horese Comics <NA>

semi\_join(superheroes,publishers) #Y의 여집합

## Joining, by = "publisher"

## # A tibble: 6 x 4  
## name alignment gender publisher  
## <chr> <chr> <chr> <chr>   
## 1 Magneto bad male Marvel   
## 2 Storm good female Marvel   
## 3 Mystique bad female Marvel   
## 4 Batman good male DC   
## 5 Joker bad male DC   
## 6 Catwomer bad female DC

# 4.margrittr

* magrittr 패키지는 연산자(operator)들의 집합들을 제공합니다.
* 데이터 연산을 왼쪽에서 오른쪽 순서로 구조화,
* nested함수 호출을 피함,
* 지역 변수 및 함수의 정의의 필요성을 최소화,
* 연산 순서 내에서 어디서나 추가 step을 만들 수 있음
* f(x)를 x%>%f()로 대체할 수 있음
* 이 연산자가 main operator(chaining)인데 해당 기능이 의미없어 보이겠지만 여러가지 기능을 결합하여 사용할때 이점이 명호가하다 \*dplyr을 불러오면 자동으로 불러와지게 된다

## 4.1 main operator (Chaining; %>%)

* 여러단계의 함수나 연산을 연결하여 한번에 수행할 때 사용
* 앞의 함수의 결과는 바로 뒤에오는 함수의 입력값이 됨
* 데이터를 여러객체에 할당하지 않아도 되기때문에 메모리 관리에 유리함

# 체인연산 사용하지 않을때

a1 <- dplyr::group\_by(flights, year,month,day)  
a2 <- select(a1,year:day,arr\_delay)  
a3 <- summarise(a2,arr= mean(arr\_delay,na.rm=TRUE))  
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 사용했을때

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

## 4.2. .의 역할

* “.”의 역할에 대해서 알아봅시다.
* 일반적으로 %>%연산자만 사용하시게 되면 제일 첫 인수에 자동으로 배정이 됩니다.

head(iris,3)

## 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  
## 3 4.7 3.2 1.3 0.2 setosa

iris %>% head(3) # = head(.,3)

## 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  
## 3 4.7 3.2 1.3 0.2 setosa

*데이터를 넘겨줘야 할 인수의 위치가 첫번째가 아닐 경우 다음과 같은 에러를 확인할 수 있음*  gsub()는 찾아 바꾸는 함수로써, 사용방법은 gusb(찾을 문자나 숫자,바꿀 문자나 숫자,데이터)

a<-c("bannananana","an apple")  
gsub("n","l",a)

## [1] "ballalalala" "al apple"

a %>% gsub("n","l")

## Warning in gsub(., "n", "l"): 인자 'pattern'는 반드시 길이가 1 보다 커야 하고,  
## 오로지 첫번째 요소만이 사용될 것입니다

## [1] "l"

a %>% gsub("n","l")

## Warning in gsub(., "n", "l"): 인자 'pattern'는 반드시 길이가 1 보다 커야 하고,  
## 오로지 첫번째 요소만이 사용될 것입니다

## [1] "l"

gsub("n","l",a)

## [1] "ballalalala" "al apple"

a %>% gsub("n","l",.)

## [1] "ballalalala" "al apple"

## 4.3 Chaining 예제

### 4.3.1 matcars aggregate

library(magrittr)

## Warning: package 'magrittr' was built under R version 3.6.3

##   
## Attaching package: 'magrittr'

## The following object is masked from 'package:purrr':  
##   
## set\_names

## The following object is masked from 'package:tidyr':  
##   
## extract

car\_data <-  
 mtcars %>% #1  
 subset(hp>100) %>% #2  
 aggregate(.~cyl, data=., FUN= . %>% mean %>% round(2)) %>% #3  
 transform(kpl = mpg %>% multiply\_by(0.4251)) %>% #4  
 print #5

## cyl mpg disp hp drat wt qsec vs am gear carb kpl  
## 1 4 25.90 108.05 111.00 3.94 2.15 17.75 1.00 1.00 4.50 2.00 11.010090  
## 2 6 19.74 183.31 122.29 3.59 3.12 17.98 0.57 0.43 3.86 3.43 8.391474  
## 3 8 15.10 353.10 209.21 3.23 4.00 16.77 0.00 0.14 3.29 3.50 6.419010

car\_data <-  
 transform(aggregate(.~cyl,  
 data=subset(mtcars,hp>100),  
 FUN =function(x) round(mean(x),2)),  
 kpl = mpg\*0.4251)  
car\_data

## cyl mpg disp hp drat wt qsec vs am gear carb kpl  
## 1 4 25.90 108.05 111.00 3.94 2.15 17.75 1.00 1.00 4.50 2.00 11.010090  
## 2 6 19.74 183.31 122.29 3.59 3.12 17.98 0.57 0.43 3.86 3.43 8.391474  
## 3 8 15.10 353.10 209.21 3.23 4.00 16.77 0.00 0.14 3.29 3.50 6.419010

### 4.3.2.예제 변환

* 2.1. 예제 tidyr의 함수들도 chaining 연산과 함께 사용하면 직관적으로 사용할 수 있습니다.

cases %>% gather(Year, n, 2:4)

## country Year n  
## 1 FR 2011 7000  
## 2 DE 2011 5800  
## 3 US 2011 15000  
## 4 FR 2012 6900  
## 5 DE 2012 6000  
## 6 US 2012 14000  
## 7 FR 2013 7000  
## 8 DE 2013 6200  
## 9 US 2013 13000

* 3.7. 예제 dplyr에서도 함께 쓰여 데이트럴 그룹화하고 수치를 요약하는 등의 작업에 특화돼있습니다.

#비행기별 비행회수, 비행거리평균, 연착시간평균 산출  
flights %>%  
 group\_by(tailnum) %>%  
 summarise(  
 count = n(),  
 dist = mean(distance,na.rm = TRUE),  
 delay = mean(arr\_delay,na.rm = TRUE)  
 )

## # A tibble: 4,044 x 4  
## tailnum count dist delay  
## <chr> <int> <dbl> <dbl>  
## 1 D942DN 4 854. 31.5   
## 2 N0EGMQ 371 676. 9.98   
## 3 N10156 153 758. 12.7   
## 4 N102UW 48 536. 2.94   
## 5 N103US 46 535. -6.93   
## 6 N104UW 47 535. 1.80   
## 7 N10575 289 520. 20.7   
## 8 N105UW 45 525. -0.267  
## 9 N107US 41 529. -5.73   
## 10 N108UW 60 534. -1.25   
## # ... with 4,034 more rows

# 5. tibble

* tibble은 tidyverse 생태계에서 데이터프레임을 대신하여 편리한 기능들 및 동작을 포함한 자료형입니다.
* factor 자동 변환
* 일부값만 출력
* 출력시 자료형 명시
* 데이터 프레임과 비교

#생성 : data.frame()  
#강제변환(Coercion) : as.data.frame()  
#데이터 불러오기 : read.\*()

## 5.1. tibble 생성

tibble()

tibble(  
 x =1:5,  
 y =1,  
 z = x^2+y  
)

## # A tibble: 5 x 3  
## x y z  
## <int> <dbl> <dbl>  
## 1 1 1 2  
## 2 2 1 5  
## 3 3 1 10  
## 4 4 1 17  
## 5 5 1 26

tribble() \* 코드 단계에서 데이터를 입력받도록 하기 위해 존재하는 함수입니다.

tribble(  
 ~x,~y,~z,  
 #---/--/----  
 "a",2,3.6,  
 "b",1,8.5  
)

## # A tibble: 2 x 3  
## x y z  
## <chr> <dbl> <dbl>  
## 1 a 2 3.6  
## 2 b 1 8.5

as\_tibble() \* 기존의 데이터 프레임을 tibble형으로 전환합니다.

iris\_tibble<- as\_tibble(iris) #기존의 데이터프레임을 tibble로  
  
print(class(iris)) #기존의 데이터 프레임 클래스

## [1] "data.frame"

print(class(iris\_tibble)) #새롭게 정의된 tibble 클래스 (데이터 프레임도)

## [1] "tbl\_df" "tbl" "data.frame"

head(iris\_tibble)

## # A tibble: 6 x 5  
## Sepal.Length Sepal.Width Petal.Length Petal.Width Species  
## <dbl> <dbl> <dbl> <dbl> <fct>   
## 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

## 5.2. 데이터 불러오기

* 데이터를 읽어올 때, dataframe이 아닌 tibble로 읽어오기 위해서,동일한 tidyverse 생태계에 속한 readr 패키지의 함수들을 필요로합니다. 이미 tidyverse를 library하였으므로 바로 이용 가능합니다.

### read\_csv(file)

\*기존의 데이터 불러오기와 동일하게 파일명을 지정하여 해당 파일을 tibble로 읽어올 수 있습니다.

read\_csv("traffic.csv")

## Warning: Missing column names filled in: 'X1' [1]

## Parsed with column specification:  
## cols(  
## X1 = col\_double(),  
## rpt.id = col\_character(),  
## rpt.contents = col\_character(),  
## info.tp = col\_character(),  
## info.tit = col\_character(),  
## occ.dtime = col\_date(format = ""),  
## reg.dtime = col\_time(format = ""),  
## end.dtime = col\_date(format = ""),  
## start.pos.x = col\_double(),  
## start.pos.y = col\_double(),  
## end.pos.x = col\_double(),  
## end.pos.y = col\_double()  
## )

## # A tibble: 500 x 12  
## X1 rpt.id rpt.contents info.tp info.tit occ.dtime reg.dtime end.dtime   
## <dbl> <chr> <chr> <chr> <chr> <date> <time> <date>   
## 1 1 01149~ 서부간선도로 안양방면~ A4 단순정보 2014-06-15 17:17 2014-06-15  
## 2 2 01149~ 동부간선도로 성수대교~ A4 단순정보 2014-06-15 17:16 2014-06-15  
## 3 3 01149~ 북부간선도로 종암분기~ A4 단순정보 2014-06-15 17:16 2014-06-15  
## 4 4 01149~ 올림픽대로 공항방면 ~ A4 단순정보 2014-06-15 17:15 2014-06-15  
## 5 5 01149~ 강변북로 (일산 → ~ A1 단순사고 2014-06-15 17:15 2014-06-15  
## 6 6 01149~ 내부순환로 성수대교방~ A4 단순정보 2014-06-15 17:14 2014-06-15  
## 7 7 01149~ 평택-시흥간고속도로 ~ A4 단순정보 2014-06-15 17:14 2014-06-15  
## 8 8 01149~ 서울-춘천간고속도로 ~ A4 단순정보 2014-06-15 17:13 2014-06-15  
## 9 9 01149~ 천안-논산간고속도로 ~ A4 단순정보 2014-06-15 17:13 2014-06-15  
## 10 10 01149~ 영동고속도로 강릉방면~ A4 단순정보 2014-06-15 17:12 2014-06-15  
## # ... with 490 more rows, and 4 more variables: start.pos.x <dbl>,  
## # start.pos.y <dbl>, end.pos.x <dbl>, end.pos.y <dbl>

### read\_csv(csv\_url)

* 외부에서 공개된 csv파일도 바로 읽어올 수 있습니다. \*\* github,gist,google drive

file\_url <- "https://gist.githubusercontent.com/theoroe3/8bc989b644adc24117bc66f50c292fc8/raw/f677a2ad811a9854c9d174178b0585a87569af60/tibbles\_data.csv"  
read\_csv(file\_url)

## Parsed with column specification:  
## cols(  
## `<-` = col\_double(),  
## `8` = col\_double(),  
## `%` = col\_double(),  
## name = col\_character()  
## )

## # A tibble: 4 x 4  
## `<-` `8` `%` name   
## <dbl> <dbl> <dbl> <chr>  
## 1 1 2 0.25 t   
## 2 2 4 0.25 h   
## 3 3 6 0.25 e   
## 4 4 8 0.25 o

### locale 설정 \* 한글이 포합된 데이터를 읽어올 떄, read.csv에서 fileEncoding으로 조정을 하였습니다. \* read\_csv 에서는 주로 locale 인자를 설정해 주어야하는데, 통상적으로 local(‘ko’,encoding=‘euc-kr’)와 같이 설정해줍니다. \* 예제는 아래의 연습문제에서 데이터를 불러오는것으로 알아보겠습니다.

# PR12 연습문제

## 문제1

* 다음은 광주광역시의 연도별 폐기물 발생현황입니다.

waste <-read\_csv("광주광역시\_연도별 폐기물 발생현황\_20171231.csv",locale=locale('ko',encoding='euc-kr'))

## Parsed with column specification:  
## cols(  
## 대분류 = col\_character(),  
## 중분류 = col\_character(),  
## 소분류 = col\_character(),  
## `2008년` = col\_double(),  
## `2009년` = col\_double(),  
## `2010년` = col\_double(),  
## `2011년` = col\_double(),  
## `2012년` = col\_double(),  
## `2013년` = col\_double(),  
## `2014년` = col\_double(),  
## `2015년` = col\_double(),  
## `2016년` = col\_double(),  
## `2017년` = col\_double()  
## )

waste

## # A tibble: 22 x 13  
## 대분류 중분류 소분류 `2008년` `2009년` `2010년` `2011년` `2012년` `2013년`  
## <chr> <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 생활폐기물~ 가정생활폐~ 매립 273. 255 235. 206. 209. 201.   
## 2 생활폐기물~ 가정생활폐~ 소각 246. 247. 232. 203 234. 242.   
## 3 생활폐기물~ 가정생활폐~ 재활용 805. 801. 771. 745. 726. 752   
## 4 생활폐기물~ 사업장생활~ 매립 21.8 35 31.4 32.1 29.6 28.8  
## 5 생활폐기물~ 사업장생활~ 소각 40.2 20.8 17.9 18.8 29.8 29   
## 6 생활폐기물~ 사업장생활~ 재활용 95.3 71.2 109. 129. 100. 73.6  
## 7 사업장배출~ 사업장배출~ 매립 163. 161. 230. 237. 213. 206.   
## 8 사업장배출~ 사업장배출~ 소각 57.4 48.5 53.1 47.8 37.2 40.6  
## 9 사업장배출~ 사업장배출~ 재활용 415. 370 422. 437. 490. 504.   
## 10 사업장배출~ 사업장배출~ 해역배출~ 10.1 107. 63.2 70.6 5.3 6   
## # ... with 12 more rows, and 4 more variables: `2014년` <dbl>, `2015년` <dbl>,  
## # `2016년` <dbl>, `2017년` <dbl>

* gather를 사용하여 waste를 아래와 같은 long 데이터 포맷으로 나타내보세요

gather(waste,Year,n,4:13)

## # A tibble: 220 x 5  
## 대분류 중분류 소분류 Year n  
## <chr> <chr> <chr> <chr> <dbl>  
## 1 생활폐기물 가정생활폐기물 매립 2008년 273.   
## 2 생활폐기물 가정생활폐기물 소각 2008년 246.   
## 3 생활폐기물 가정생활폐기물 재활용 2008년 805.   
## 4 생활폐기물 사업장생활계폐기물 매립 2008년 21.8  
## 5 생활폐기물 사업장생활계폐기물 소각 2008년 40.2  
## 6 생활폐기물 사업장생활계폐기물 재활용 2008년 95.3  
## 7 사업장배출시설계폐기물 사업장배출시설계폐기물 매립 2008년 163.   
## 8 사업장배출시설계폐기물 사업장배출시설계폐기물 소각 2008년 57.4  
## 9 사업장배출시설계폐기물 사업장배출시설계폐기물 재활용 2008년 415.   
## 10 사업장배출시설계폐기물 사업장배출시설계폐기물 해역배출 2008년 10.1  
## # ... with 210 more rows

## 문제2

\*다음은 서울시 공공자전거 따릉이의 2019년 5월 대여에 관한 정보입니다.

bike\_tibble <- read\_csv("bike\_sample.csv")

## Parsed with column specification:  
## cols(  
## 대여일자 = col\_date(format = ""),  
## 대여시간 = col\_double(),  
## 대여소번호 = col\_double(),  
## 대여소명 = col\_character(),  
## 대여구분코드 = col\_character(),  
## 성별 = col\_character(),  
## 연령대코드 = col\_character(),  
## 이용건수 = col\_double(),  
## 운동량 = col\_character(),  
## 탄소량 = col\_character(),  
## 이동거리 = col\_double(),  
## 이동시간 = col\_double()  
## )

* 해당 데이터에서 다음과 같이 연령대별 정보를 요약해보세요.

by\_age <- bike\_tibble %>%  
 group\_by(연령대코드) %>%  
 summarize(count=n(),rentalmean=mean(이용건수))  
  
print(by\_age)

## # A tibble: 7 x 3  
## 연령대코드 count rentalmean  
## <chr> <int> <dbl>  
## 1 ~10대 24 1.25  
## 2 20대 209 1.48  
## 3 30대 143 1.24  
## 4 40대 74 1.12  
## 5 50대 36 1.08  
## 6 60대 11 1   
## 7 70대~ 3 1

## 문제3

* raw.csv는 국회의원 별 의원비 지출내역이고, join.csv는 국회의원 명단입니다.

raw\_data <- read\_csv("raw.csv", locale=locale('ko',encoding='euc-kr'))

## Warning: Missing column names filled in: 'X1' [1]

## Parsed with column specification:  
## cols(  
## X1 = col\_double(),  
## total\_num = col\_double(),  
## cong\_num = col\_double(),  
## name = col\_character(),  
## party = col\_character(),  
## region = col\_character(),  
## date = col\_date(format = ""),  
## item = col\_character(),  
## expense = col\_double(),  
## store = col\_character(),  
## category = col\_character(),  
## region2 = col\_character(),  
## date2 = col\_date(format = ""),  
## date\_month = col\_character()  
## )

join\_data <-read\_csv("join.csv",locale=locale("ko",encoding='euc-kr'))

## Warning: Missing column names filled in: 'X1' [1], 'X16' [16], 'X17' [17],  
## 'X18' [18], 'X19' [19]

## Warning: Duplicated column names deduplicated: 'achievement' =>  
## 'achievement\_1' [15]

## Parsed with column specification:  
## cols(  
## X1 = col\_double(),  
## region = col\_character(),  
## district = col\_character(),  
## party = col\_character(),  
## name = col\_character(),  
## gender = col\_character(),  
## birth = col\_date(format = ""),  
## job = col\_character(),  
## achievement = col\_character(),  
## career = col\_character(),  
## vote\_rate = col\_number(),  
## age = col\_double(),  
## regeion = col\_character(),  
## job2 = col\_character(),  
## achievement\_1 = col\_character(),  
## X16 = col\_character(),  
## X17 = col\_character(),  
## X18 = col\_character(),  
## X19 = col\_double()  
## )

raw\_data ; names(raw\_data)

## # A tibble: 146,224 x 14  
## X1 total\_num cong\_num name party region date item expense store  
## <dbl> <dbl> <dbl> <chr> <chr> <chr> <date> <chr> <dbl> <chr>  
## 1 1 217616 309 현영희~ 무소속~ 비례 2014-01-03 항공료~ 68600 대한항공~  
## 2 2 217617 309 현영희~ 무소속~ 비례 2014-01-03 전문가 ~ 78000 카페모차~  
## 3 3 217618 309 현영희~ 무소속~ 비례 2014-01-03 항공료~ 97100 대한항공~  
## 4 4 217619 309 현영희~ 무소속~ 비례 2014-01-03 유류비~ 138000 예스제이~  
## 5 5 217620 309 현영희~ 무소속~ 비례 2014-01-05 유류비~ 60000 과정로셀~  
## 6 6 221930 1 강기윤~ 새누리당~ 경남 창원~ 2014-01-02 우편발송~ 3070 창원우체~  
## 7 7 221931 1 강기윤~ 새누리당~ 경남 창원~ 2014-01-03 식대 111000 초가집식~  
## 8 8 221932 1 강기윤~ 새누리당~ 경남 창원~ 2014-01-04 차량 L~ 61254 광신개발~  
## 9 9 221933 1 강기윤~ 새누리당~ 경남 창원~ 2014-01-06 커피 구~ 30000 GS25~  
## 10 10 221934 1 강기윤~ 새누리당~ 경남 창원~ 2014-01-06 식대 88000 일호추어~  
## # ... with 146,214 more rows, and 4 more variables: category <chr>,  
## # region2 <chr>, date2 <date>, date\_month <chr>

## [1] "X1" "total\_num" "cong\_num" "name" "party"   
## [6] "region" "date" "item" "expense" "store"   
## [11] "category" "region2" "date2" "date\_month"

join\_data ; names(join\_data)

## # A tibble: 300 x 19  
## X1 region district party name gender birth job achievement career  
## <dbl> <chr> <chr> <chr> <chr> <chr> <date> <chr> <chr> <chr>   
## 1 94 비례 비례대표 새누리당~ 민현주~ 여 1969-07-23 경기대학~ Cornell 대학~ (현)경기~  
## 2 135 비례 비례대표 새누리당~ 신경림~ 여 1954-03-22 이화여자~ Teachers C~ (현)한국~  
## 3 235 경기 하남시 새누리당~ 이현재~ 남 1949-04-25 정당인~ 건국대학교 대학원 ~ (전)중소~  
## 4 294 인천 남구갑 새누리당~ 홍일표~ 남 1956-02-11 국회의원~ 건국대학교 대학원 ~ (전)인천~  
## 5 185 충북 충주시 새누리당~ 윤진식~ 남 1946-03-04 국회의원~ 건국대학교 일반대학~ 전)산업자~  
## 6 131 비례 비례대표 새누리당~ 손인춘~ 여 1959-05-13 (주)인~ 건국대학교 일반대학~ (현)(주~  
## 7 69 서울 광진구갑 민주통합~ 김한길~ 남 1953-09-17 정당인~ 건국대학교 정치외교~ (전)문화~  
## 8 122 경기 광명시갑 민주통합~ 백재현~ 남 1951-07-04 국회의원~ 경기대학교 무역학과~ (현) 민~  
## 9 4 전북 남원시순창군~ 통합진보~ 강동원~ 남 1953-01-20 정당인~ 경기대학교 정치전문~ (전)농수~  
## 10 192 서울 노원구갑 새누리당~ 이노근~ 남 1954-03-09 광운대학~ 경기대학교 정치전문~ (전)노원~  
## # ... with 290 more rows, and 9 more variables: vote\_rate <dbl>, age <dbl>,  
## # regeion <chr>, job2 <chr>, achievement\_1 <chr>, X16 <chr>, X17 <chr>,  
## # X18 <chr>, X19 <dbl>

## [1] "X1" "region" "district" "party"   
## [5] "name" "gender" "birth" "job"   
## [9] "achievement" "career" "vote\_rate" "age"   
## [13] "regeion" "job2" "achievement\_1" "X16"   
## [17] "X17" "X18" "X19"

* 두 데이터를 조인하여 의원비 지출이 가장 많은 10명의 직업과 학력을 확인하세요.
* HINT1 : 두 데이터를 조인하기 위해서는 raw.csv의 데이터를 group\_by(),summarise()함수 또는 aggregate()를 사용하여 요약해야함.
* HINT2 : 이름을 기준으로 join할것

processed\_data <- raw\_data %>%  
 group\_by(name,party) %>%  
 summarise(expense\_sum=sum(expense)) %>%  
 arrange(desc(expense\_sum)) %>%  
 head(10) %>%  
 merge(join\_data,by='name') %>%  
 arrange(desc(expense\_sum)) %>%  
 head(10) %>%  
 select(c(name,party.x,expense\_sum,job,achievement)) %>%  
 as\_tibble() %>%  
 mutate(job = as.character(job)) %>%  
 mutate(achievement = as.character(achievement)) %>%  
 mutate(party.x = as.character(party.x)) %>%  
 mutate(name = as.character(name))  
  
  
colnames(processed\_data) <-c('name','party','expense\_sum','job','achievement')  
  
print(processed\_data)

## # A tibble: 10 x 5  
## name party expense\_sum job achievement   
## <chr> <chr> <dbl> <chr> <chr>   
## 1 이명수 새누리당~ 446913569 국회의원 성균관대 대학원 졸업(행정학과 행정학 박사)~  
## 2 김상민 새누리당~ 402282444 대학생자원봉사단 V원정대 대표~ 아주대학교 사학과졸업   
## 3 조원진 새누리당~ 392767643 국회의원 영남대학교 행정대학원 정책분석학과 졸업(행정학 석사)~  
## 4 서상기 새누리당~ 351375323 국회의원 美드렉셀대학교 공학박사(1972.09~1976.06)~  
## 5 이상규 통합진보~ 346551904 정당인 서울대학교 법과대학 공법학과 졸업~  
## 6 심상정 진보정의~ 336436592 정당인 서울대학교 사회교육과 졸업   
## 7 박지원 새정치 331136715 국회의원 단국대 상학과 졸업   
## 8 정병국 새누리당~ 330634706 국회의원 성균관 대학교 대학원 졸업(정치학박사)~  
## 9 유성엽 새정치 330615390 국회의원 서울대학교 사회과학대학 외교학과 졸업~  
## 10 이종걸 새정치 323533338 변호사 서울대학교 법과대학 공법학과 졸업~