dplyr

박효선 1585063

2018년 11월 30일

###############################################  
###############################################  
# The easiest way to get dplyr is to install the whole tidyverse:  
#install.packages("tidyverse")  
  
# Alternatively, install just dplyr:  
#install.packages("dplyr")  
  
# install.packages("devtools")  
# devtools::install\_github("tidyverse/dplyr")  
  
###############################################  
###############################################  
#dplyr 함수들  
  
# 행(row)선별 : filter(), slice()  
# 행(row)정렬 : arrange()  
# 열(column)선별 : select()  
# 열(column)조건부 선별 :   
# select(df, starts\_with()),  
# select(df, ends\_with()),  
# select(df, contains())  
# select(df, matchs()),   
# select(df, one\_of()),  
# select(df, run\_range())  
# 변수이름변경 : rename()  
#   
# mutate() : 새로운 변수를 추가한다. 기존변수 + 새로운 변수   
# summarise() : 변수요약. mutiple values -> a single summary.  
# arrange() : 행 정렬  
  
  
  
###############################################  
###############################################  
  
#Error in select() : unused arguments()  
#MASS 패키지의 select() 와 dplyr 패키지의 select()  
  
#충돌해결 : select 는 dplyr 패키지의 select 임을 명시적으로 지저  
select <- dplyr::select  
starwars %>%   
 select(name, ends\_with("color"))

## # A tibble: 87 x 4  
## name hair\_color skin\_color eye\_color  
## <chr> <chr> <chr> <chr>   
## 1 Luke Skywalker blond fair blue   
## 2 C-3PO <NA> gold yellow   
## 3 R2-D2 <NA> white, blue red   
## 4 Darth Vader none white yellow   
## 5 Leia Organa brown light brown   
## 6 Owen Lars brown, grey light blue   
## 7 Beru Whitesun lars brown light blue   
## 8 R5-D4 <NA> white, red red   
## 9 Biggs Darklighter black light brown   
## 10 Obi-Wan Kenobi auburn, white fair blue-gray  
## # ... with 77 more rows

starwars %>%   
 mutate(name, bmi = mass / ((height / 100) ^ 2)) %>%  
 select(name:mass, bmi)

## Warning: package 'bindrcpp' was built under R version 3.5.1

## # A tibble: 87 x 4  
## name height mass bmi  
## <chr> <int> <dbl> <dbl>  
## 1 Luke Skywalker 172 77 26.0  
## 2 C-3PO 167 75 26.9  
## 3 R2-D2 96 32 34.7  
## 4 Darth Vader 202 136 33.3  
## 5 Leia Organa 150 49 21.8  
## 6 Owen Lars 178 120 37.9  
## 7 Beru Whitesun lars 165 75 27.5  
## 8 R5-D4 97 32 34.0  
## 9 Biggs Darklighter 183 84 25.1  
## 10 Obi-Wan Kenobi 182 77 23.2  
## # ... with 77 more rows

starwars %>%   
 arrange(desc(mass))

## # A tibble: 87 x 13  
## name height mass hair\_color skin\_color eye\_color birth\_year gender  
## <chr> <int> <dbl> <chr> <chr> <chr> <dbl> <chr>   
## 1 Jabba ~ 175 1358 <NA> green-tan~ orange 600 herma~  
## 2 Grievo~ 216 159 none brown, wh~ green, y~ NA male   
## 3 IG-88 200 140 none metal red 15 none   
## 4 Darth ~ 202 136 none white yellow 41.9 male   
## 5 Tarfful 234 136 brown brown blue NA male   
## 6 Owen L~ 178 120 brown, grey light blue 52 male   
## 7 Bossk 190 113 none green red 53 male   
## 8 Chewba~ 228 112 brown unknown blue 200 male   
## 9 Jek To~ 180 110 brown fair blue NA male   
## 10 Dexter~ 198 102 none brown yellow NA male   
## # ... with 77 more rows, and 5 more variables: homeworld <chr>,  
## # species <chr>, films <list>, vehicles <list>, starships <list>

starwars %>%  
 group\_by(species) %>%  
 summarise(  
 n = n(),  
 mass = mean(mass, na.rm = TRUE)  
 ) %>%  
 filter(n > 1)

## # A tibble: 9 x 3  
## species n mass  
## <chr> <int> <dbl>  
## 1 Droid 5 69.8  
## 2 Gungan 3 74   
## 3 Human 35 82.8  
## 4 Kaminoan 2 88   
## 5 Mirialan 2 53.1  
## 6 Twi'lek 2 55   
## 7 Wookiee 2 124   
## 8 Zabrak 2 80   
## 9 <NA> 5 48

###############################################  
###############################################  
#dplyr 기본문법들  
  
###############################################  
#데이터 프레임  
#subset Cars93  
Cars93\_1<- Cars93[,1:8]  
str(Cars93\_1)

## 'data.frame': 93 obs. of 8 variables:  
## $ Manufacturer: Factor w/ 32 levels "Acura","Audi",..: 1 1 2 2 3 4 4 4 4 5 ...  
## $ Model : Factor w/ 93 levels "100","190E","240",..: 49 56 9 1 6 24 54 74 73 35 ...  
## $ Type : Factor w/ 6 levels "Compact","Large",..: 4 3 1 3 3 3 2 2 3 2 ...  
## $ Min.Price : num 12.9 29.2 25.9 30.8 23.7 14.2 19.9 22.6 26.3 33 ...  
## $ Price : num 15.9 33.9 29.1 37.7 30 15.7 20.8 23.7 26.3 34.7 ...  
## $ Max.Price : num 18.8 38.7 32.3 44.6 36.2 17.3 21.7 24.9 26.3 36.3 ...  
## $ MPG.city : int 25 18 20 19 22 22 19 16 19 16 ...  
## $ MPG.highway : int 31 25 26 26 30 31 28 25 27 25 ...

#number of cars by Type  
table(Cars93\_1$Type)

##   
## Compact Large Midsize Small Sporty Van   
## 16 11 22 21 14 9

#########################  
#데이터 프레임의 행(row) 부분집합 선별 : filter(), slice()  
  
#filter(dataframe, filter condition1, filter condition2, ....) : &(AND)조건으로 row 데이터 부분집합 선별  
#차종(type)이 "compact"이고 & 최대가격(max.price)이 20백$ 이하이고 & 고속도로 연비(MPG.highway) 가 30 이상인 관측치를 선별  
Cars93\_1 %>% filter( Type == c("Compact"), Max.Price <= 20, MPG.highway >= 30)

## Manufacturer Model Type Min.Price Price Max.Price MPG.city  
## 1 Chevrolet Cavalier Compact 8.5 13.4 18.3 25  
## 2 Chevrolet Corsica Compact 11.4 11.4 11.4 25  
## 3 Mazda 626 Compact 14.3 16.5 18.7 26  
## 4 Nissan Altima Compact 13.0 15.7 18.3 24  
## 5 Oldsmobile Achieva Compact 13.0 13.5 14.0 24  
## 6 Pontiac Sunbird Compact 9.4 11.1 12.8 23  
## MPG.highway  
## 1 36  
## 2 34  
## 3 34  
## 4 30  
## 5 31  
## 6 31

#filter(dataframe, filter condition1 | filter condition2 | ....) : |(OR)조건으로 row 데이터 부분집합 선별  
#차종(Type)이 "Compact"이거나 | 최대가격(Max.Price)이 20 백$ 이하이거나| 고속도로 연비(MPG.highway) 가 30 이상인 관측치를 선별  
Cars93\_1 %>% filter( Type == c("Compact") | Max.Price <= 20 | MPG.highway >= 30)

## Manufacturer Model Type Min.Price Price Max.Price MPG.city  
## 1 Acura Integra Small 12.9 15.9 18.8 25  
## 2 Audi 90 Compact 25.9 29.1 32.3 20  
## 3 BMW 535i Midsize 23.7 30.0 36.2 22  
## 4 Buick Century Midsize 14.2 15.7 17.3 22  
## 5 Chevrolet Cavalier Compact 8.5 13.4 18.3 25  
## 6 Chevrolet Corsica Compact 11.4 11.4 11.4 25  
## 7 Chevrolet Camaro Sporty 13.4 15.1 16.8 19  
## 8 Chevrolet Lumina Midsize 13.4 15.9 18.4 21  
## 9 Chevrolet Lumina\_APV Van 14.7 16.3 18.0 18  
## 10 Chevrolet Astro Van 14.7 16.6 18.6 15  
## 11 Chevrolet Caprice Large 18.0 18.8 19.6 17  
## 12 Chrylser Concorde Large 18.4 18.4 18.4 20  
## 13 Chrysler LeBaron Compact 14.5 15.8 17.1 23  
## 14 Dodge Colt Small 7.9 9.2 10.6 29  
## 15 Dodge Shadow Small 8.4 11.3 14.2 23  
## 16 Dodge Spirit Compact 11.9 13.3 14.7 22  
## 17 Dodge Dynasty Midsize 14.8 15.6 16.4 21  
## 18 Eagle Summit Small 7.9 12.2 16.5 29  
## 19 Ford Festiva Small 6.9 7.4 7.9 31  
## 20 Ford Escort Small 8.4 10.1 11.9 23  
## 21 Ford Tempo Compact 10.4 11.3 12.2 22  
## 22 Ford Probe Sporty 12.8 14.0 15.2 24  
## 23 Ford Taurus Midsize 15.6 20.2 24.8 21  
## 24 Geo Metro Small 6.7 8.4 10.0 46  
## 25 Geo Storm Sporty 11.5 12.5 13.5 30  
## 26 Honda Prelude Sporty 17.0 19.8 22.7 24  
## 27 Honda Civic Small 8.4 12.1 15.8 42  
## 28 Honda Accord Compact 13.8 17.5 21.2 24  
## 29 Hyundai Excel Small 6.8 8.0 9.2 29  
## 30 Hyundai Elantra Small 9.0 10.0 11.0 22  
## 31 Hyundai Scoupe Sporty 9.1 10.0 11.0 26  
## 32 Hyundai Sonata Midsize 12.4 13.9 15.3 20  
## 33 Mazda 323 Small 7.4 8.3 9.1 29  
## 34 Mazda Protege Small 10.9 11.6 12.3 28  
## 35 Mazda 626 Compact 14.3 16.5 18.7 26  
## 36 Mercedes-Benz 190E Compact 29.0 31.9 34.9 20  
## 37 Mercury Capri Sporty 13.3 14.1 15.0 23  
## 38 Mercury Cougar Midsize 14.9 14.9 14.9 19  
## 39 Mitsubishi Mirage Small 7.7 10.3 12.9 29  
## 40 Nissan Sentra Small 8.7 11.8 14.9 29  
## 41 Nissan Altima Compact 13.0 15.7 18.3 24  
## 42 Oldsmobile Achieva Compact 13.0 13.5 14.0 24  
## 43 Oldsmobile Cutlass\_Ciera Midsize 14.2 16.3 18.4 23  
## 44 Oldsmobile Silhouette Van 19.5 19.5 19.5 18  
## 45 Plymouth Laser Sporty 11.4 14.4 17.4 23  
## 46 Pontiac LeMans Small 8.2 9.0 9.9 31  
## 47 Pontiac Sunbird Compact 9.4 11.1 12.8 23  
## 48 Saab 900 Compact 20.3 28.7 37.1 20  
## 49 Saturn SL Small 9.2 11.1 12.9 28  
## 50 Subaru Justy Small 7.3 8.4 9.5 33  
## 51 Subaru Loyale Small 10.5 10.9 11.3 25  
## 52 Subaru Legacy Compact 16.3 19.5 22.7 23  
## 53 Suzuki Swift Small 7.3 8.6 10.0 39  
## 54 Toyota Tercel Small 7.8 9.8 11.8 32  
## 55 Toyota Celica Sporty 14.2 18.4 22.6 25  
## 56 Volkswagen Fox Small 8.7 9.1 9.5 25  
## 57 Volkswagen Passat Compact 17.6 20.0 22.4 21  
## 58 Volvo 240 Compact 21.8 22.7 23.5 21  
## MPG.highway  
## 1 31  
## 2 26  
## 3 30  
## 4 31  
## 5 36  
## 6 34  
## 7 28  
## 8 29  
## 9 23  
## 10 20  
## 11 26  
## 12 28  
## 13 28  
## 14 33  
## 15 29  
## 16 27  
## 17 27  
## 18 33  
## 19 33  
## 20 30  
## 21 27  
## 22 30  
## 23 30  
## 24 50  
## 25 36  
## 26 31  
## 27 46  
## 28 31  
## 29 33  
## 30 29  
## 31 34  
## 32 27  
## 33 37  
## 34 36  
## 35 34  
## 36 29  
## 37 26  
## 38 26  
## 39 33  
## 40 33  
## 41 30  
## 42 31  
## 43 31  
## 44 23  
## 45 30  
## 46 41  
## 47 31  
## 48 26  
## 49 38  
## 50 37  
## 51 30  
## 52 30  
## 53 43  
## 54 37  
## 55 32  
## 56 33  
## 57 30  
## 58 28

#silce(dataframe, from, to) : 위치를 지정해서 row 데이터 부분집합 선별하기   
Cars93\_1 %>% slice(6:10)

## Manufacturer Model Type Min.Price Price Max.Price MPG.city  
## 1 Buick Century Midsize 14.2 15.7 17.3 22  
## 2 Buick LeSabre Large 19.9 20.8 21.7 19  
## 3 Buick Roadmaster Large 22.6 23.7 24.9 16  
## 4 Buick Riviera Midsize 26.3 26.3 26.3 19  
## 5 Cadillac DeVille Large 33.0 34.7 36.3 16  
## MPG.highway  
## 1 31  
## 2 28  
## 3 25  
## 4 27  
## 5 25

##########################  
#데이터프레임 행 정렬하기 : arrange()  
#arrange(dataframe, order criterion1, order criterion 2, ...)  
 #오름차순(asecending) 정렬: 기본정렬 옵션  
 #내림차순(descending) 정렬: desc()  
  
#고속도로 연비(MPG.highway) 가 높은 순서대로 정렬을 하시오. 만약 고속도로 연비가 동일하다면 최고가격(Max.Price)가 낮은 순서대로 정렬하시오.  
Cars93\_1 %>% arrange(desc(MPG.highway), Max.Price) %>% head(20)

## Manufacturer Model Type Min.Price Price Max.Price MPG.city  
## 1 Geo Metro Small 6.7 8.4 10.0 46  
## 2 Honda Civic Small 8.4 12.1 15.8 42  
## 3 Suzuki Swift Small 7.3 8.6 10.0 39  
## 4 Pontiac LeMans Small 8.2 9.0 9.9 31  
## 5 Saturn SL Small 9.2 11.1 12.9 28  
## 6 Mazda 323 Small 7.4 8.3 9.1 29  
## 7 Subaru Justy Small 7.3 8.4 9.5 33  
## 8 Toyota Tercel Small 7.8 9.8 11.8 32  
## 9 Mazda Protege Small 10.9 11.6 12.3 28  
## 10 Geo Storm Sporty 11.5 12.5 13.5 30  
## 11 Chevrolet Cavalier Compact 8.5 13.4 18.3 25  
## 12 Hyundai Scoupe Sporty 9.1 10.0 11.0 26  
## 13 Chevrolet Corsica Compact 11.4 11.4 11.4 25  
## 14 Mazda 626 Compact 14.3 16.5 18.7 26  
## 15 Ford Festiva Small 6.9 7.4 7.9 31  
## 16 Hyundai Excel Small 6.8 8.0 9.2 29  
## 17 Volkswagen Fox Small 8.7 9.1 9.5 25  
## 18 Dodge Colt Small 7.9 9.2 10.6 29  
## 19 Mitsubishi Mirage Small 7.7 10.3 12.9 29  
## 20 Nissan Sentra Small 8.7 11.8 14.9 29  
## MPG.highway  
## 1 50  
## 2 46  
## 3 43  
## 4 41  
## 5 38  
## 6 37  
## 7 37  
## 8 37  
## 9 36  
## 10 36  
## 11 36  
## 12 34  
## 13 34  
## 14 34  
## 15 33  
## 16 33  
## 17 33  
## 18 33  
## 19 33  
## 20 33

######################  
#데이터 프레임 변수(column) 선별하기 : select()  
  
#select(dataframe, VAR1, VAR2, ...) : 선별하고자 하는 변수 이름을 기입  
#Cars93\_1 데이터 프레임으로부터 제조사명(Manufacturer), 최대가격(Max.Price), 고속도로연비(MPG.highway) 3개 변수(칼럼)를 선별하시오.  
Cars93\_1 %>% select(Manufacturer, Max.Price, MPG.highway)%>% head(20)

## Manufacturer Max.Price MPG.highway  
## 1 Acura 18.8 31  
## 2 Acura 38.7 25  
## 3 Audi 32.3 26  
## 4 Audi 44.6 26  
## 5 BMW 36.2 30  
## 6 Buick 17.3 31  
## 7 Buick 21.7 28  
## 8 Buick 24.9 25  
## 9 Buick 26.3 27  
## 10 Cadillac 36.3 25  
## 11 Cadillac 42.7 25  
## 12 Chevrolet 18.3 36  
## 13 Chevrolet 11.4 34  
## 14 Chevrolet 16.8 28  
## 15 Chevrolet 18.4 29  
## 16 Chevrolet 18.0 23  
## 17 Chevrolet 18.6 20  
## 18 Chevrolet 19.6 26  
## 19 Chevrolet 41.5 25  
## 20 Chrylser 18.4 28

#select(dataframe, a:n) : a 번째부터 n 번째 변수 선별  
Cars93\_1 %>% select(Manufacturer:Price) %>% head(20)

## Manufacturer Model Type Min.Price Price  
## 1 Acura Integra Small 12.9 15.9  
## 2 Acura Legend Midsize 29.2 33.9  
## 3 Audi 90 Compact 25.9 29.1  
## 4 Audi 100 Midsize 30.8 37.7  
## 5 BMW 535i Midsize 23.7 30.0  
## 6 Buick Century Midsize 14.2 15.7  
## 7 Buick LeSabre Large 19.9 20.8  
## 8 Buick Roadmaster Large 22.6 23.7  
## 9 Buick Riviera Midsize 26.3 26.3  
## 10 Cadillac DeVille Large 33.0 34.7  
## 11 Cadillac Seville Midsize 37.5 40.1  
## 12 Chevrolet Cavalier Compact 8.5 13.4  
## 13 Chevrolet Corsica Compact 11.4 11.4  
## 14 Chevrolet Camaro Sporty 13.4 15.1  
## 15 Chevrolet Lumina Midsize 13.4 15.9  
## 16 Chevrolet Lumina\_APV Van 14.7 16.3  
## 17 Chevrolet Astro Van 14.7 16.6  
## 18 Chevrolet Caprice Large 18.0 18.8  
## 19 Chevrolet Corvette Sporty 34.6 38.0  
## 20 Chrylser Concorde Large 18.4 18.4

Cars93\_1 %>% select(1:5) %>% head(20)

## Manufacturer Model Type Min.Price Price  
## 1 Acura Integra Small 12.9 15.9  
## 2 Acura Legend Midsize 29.2 33.9  
## 3 Audi 90 Compact 25.9 29.1  
## 4 Audi 100 Midsize 30.8 37.7  
## 5 BMW 535i Midsize 23.7 30.0  
## 6 Buick Century Midsize 14.2 15.7  
## 7 Buick LeSabre Large 19.9 20.8  
## 8 Buick Roadmaster Large 22.6 23.7  
## 9 Buick Riviera Midsize 26.3 26.3  
## 10 Cadillac DeVille Large 33.0 34.7  
## 11 Cadillac Seville Midsize 37.5 40.1  
## 12 Chevrolet Cavalier Compact 8.5 13.4  
## 13 Chevrolet Corsica Compact 11.4 11.4  
## 14 Chevrolet Camaro Sporty 13.4 15.1  
## 15 Chevrolet Lumina Midsize 13.4 15.9  
## 16 Chevrolet Lumina\_APV Van 14.7 16.3  
## 17 Chevrolet Astro Van 14.7 16.6  
## 18 Chevrolet Caprice Large 18.0 18.8  
## 19 Chevrolet Corvette Sporty 34.6 38.0  
## 20 Chrylser Concorde Large 18.4 18.4

#select(dataframe, -var1, -var2, ....): ㅁa번째부터 n 번째 변수는 빼고 선별  
Cars93\_1 %>% select(-(Manufacturer:Price)) %>% head(20)

## Max.Price MPG.city MPG.highway  
## 1 18.8 25 31  
## 2 38.7 18 25  
## 3 32.3 20 26  
## 4 44.6 19 26  
## 5 36.2 22 30  
## 6 17.3 22 31  
## 7 21.7 19 28  
## 8 24.9 16 25  
## 9 26.3 19 27  
## 10 36.3 16 25  
## 11 42.7 16 25  
## 12 18.3 25 36  
## 13 11.4 25 34  
## 14 16.8 19 28  
## 15 18.4 21 29  
## 16 18.0 18 23  
## 17 18.6 15 20  
## 18 19.6 17 26  
## 19 41.5 17 25  
## 20 18.4 20 28

Cars93\_1 %>% select(-(1:5)) %>% head(20)

## Max.Price MPG.city MPG.highway  
## 1 18.8 25 31  
## 2 38.7 18 25  
## 3 32.3 20 26  
## 4 44.6 19 26  
## 5 36.2 22 30  
## 6 17.3 22 31  
## 7 21.7 19 28  
## 8 24.9 16 25  
## 9 26.3 19 27  
## 10 36.3 16 25  
## 11 42.7 16 25  
## 12 18.3 25 36  
## 13 11.4 25 34  
## 14 16.8 19 28  
## 15 18.4 21 29  
## 16 18.0 18 23  
## 17 18.6 15 20  
## 18 19.6 17 26  
## 19 41.5 17 25  
## 20 18.4 20 28

#select(datarame, starts\_with("xx\_name")) : "xx\_name"으로 시작하는 모든 변수 선별  
#Cars93\_1 데이터 프레임에서 "MPG"로 끝나는 모든 변수를 선별  
Cars93\_1 %>% select(starts\_with("MPG"))%>% head(20)

## MPG.city MPG.highway  
## 1 25 31  
## 2 18 25  
## 3 20 26  
## 4 19 26  
## 5 22 30  
## 6 22 31  
## 7 19 28  
## 8 16 25  
## 9 19 27  
## 10 16 25  
## 11 16 25  
## 12 25 36  
## 13 25 34  
## 14 19 28  
## 15 21 29  
## 16 18 23  
## 17 15 20  
## 18 17 26  
## 19 17 25  
## 20 20 28

#select(datarame, ends\_with("xx\_name")) : "xx\_name"으로 끝나는 모든 변수 선별  
#Cars93\_1 데이터 프레임에서 "Price"로 끝나는 모든 변수를 선별  
Cars93\_1 %>% select(ends\_with("Price"))%>% head(20)

## Min.Price Price Max.Price  
## 1 12.9 15.9 18.8  
## 2 29.2 33.9 38.7  
## 3 25.9 29.1 32.3  
## 4 30.8 37.7 44.6  
## 5 23.7 30.0 36.2  
## 6 14.2 15.7 17.3  
## 7 19.9 20.8 21.7  
## 8 22.6 23.7 24.9  
## 9 26.3 26.3 26.3  
## 10 33.0 34.7 36.3  
## 11 37.5 40.1 42.7  
## 12 8.5 13.4 18.3  
## 13 11.4 11.4 11.4  
## 14 13.4 15.1 16.8  
## 15 13.4 15.9 18.4  
## 16 14.7 16.3 18.0  
## 17 14.7 16.6 18.6  
## 18 18.0 18.8 19.6  
## 19 34.6 38.0 41.5  
## 20 18.4 18.4 18.4

#select(dataframe, contain("xx\_name")) : "xx\_name"을 포함하는 모든 변수 선별  
#Cars93\_1 데이터 프레임에서 "P"를 포함하는 모든 변수를 선별  
Cars93\_1 %>% select(contains("P"))%>% head(20)

## Type Min.Price Price Max.Price MPG.city MPG.highway  
## 1 Small 12.9 15.9 18.8 25 31  
## 2 Midsize 29.2 33.9 38.7 18 25  
## 3 Compact 25.9 29.1 32.3 20 26  
## 4 Midsize 30.8 37.7 44.6 19 26  
## 5 Midsize 23.7 30.0 36.2 22 30  
## 6 Midsize 14.2 15.7 17.3 22 31  
## 7 Large 19.9 20.8 21.7 19 28  
## 8 Large 22.6 23.7 24.9 16 25  
## 9 Midsize 26.3 26.3 26.3 19 27  
## 10 Large 33.0 34.7 36.3 16 25  
## 11 Midsize 37.5 40.1 42.7 16 25  
## 12 Compact 8.5 13.4 18.3 25 36  
## 13 Compact 11.4 11.4 11.4 25 34  
## 14 Sporty 13.4 15.1 16.8 19 28  
## 15 Midsize 13.4 15.9 18.4 21 29  
## 16 Van 14.7 16.3 18.0 18 23  
## 17 Van 14.7 16.6 18.6 15 20  
## 18 Large 18.0 18.8 19.6 17 26  
## 19 Sporty 34.6 38.0 41.5 17 25  
## 20 Large 18.4 18.4 18.4 20 28

#select(dataframe, one\_of(Vars)) : 변수 이름 그룹에 포함된 모든 변수 선별  
#"Manufacturer", "MAX.Price", "MPG.highway" 의 3개 변수이름을 포함하는 변수 그룹이 있다고 할 때, Cars93 데이터 프레임에서 이 변수 그룹에 있는 변수가 있다면(<- 즉, 있을 수도 있지만 없을 수도 있다는 뜻임!) 모두 선별  
vars <- c("Manufacturer", "MAX.Price", "MPG.highway")  
Cars93\_1 %>% select( one\_of(vars)) %>% head(20)

## Warning: Unknown columns: `MAX.Price`

## Manufacturer MPG.highway  
## 1 Acura 31  
## 2 Acura 25  
## 3 Audi 26  
## 4 Audi 26  
## 5 BMW 30  
## 6 Buick 31  
## 7 Buick 28  
## 8 Buick 25  
## 9 Buick 27  
## 10 Cadillac 25  
## 11 Cadillac 25  
## 12 Chevrolet 36  
## 13 Chevrolet 34  
## 14 Chevrolet 28  
## 15 Chevrolet 29  
## 16 Chevrolet 23  
## 17 Chevrolet 20  
## 18 Chevrolet 26  
## 19 Chevrolet 25  
## 20 Chrylser 28

#변수그룹 vars 에 데이터 프레임 포함된 변수는 선별해서 반환, 만약 없으면 Warning Message제시  
  
  
#######################  
#데이터 프레임 변수 이름 변경하기: rename()  
#rename(dataframe, new\_var1 = old\_var1, new\_var2 = old\_var2, ...)  
#새로운 변수 이름을 앞, 이전 변수 이름을 뒤  
#따옴표 x, 여러개이면 , 로 나열  
  
names(Cars93\_1)

## [1] "Manufacturer" "Model" "Type" "Min.Price"   
## [5] "Price" "Max.Price" "MPG.city" "MPG.highway"

Cars93\_2 <- Cars93\_1 %>% rename(New\_Manufacturer = Manufacturer,  
 New\_Model = Model,   
 New\_Type = Type,   
 New\_Min.Price = Min.Price,   
 New\_Price = Price,   
 New\_Max.Price = Max.Price,  
 New\_MPG.city = MPG.city,   
 New\_MPG.highway = MPG.highway)  
names(Cars93\_2)

## [1] "New\_Manufacturer" "New\_Model" "New\_Type"   
## [4] "New\_Min.Price" "New\_Price" "New\_Max.Price"   
## [7] "New\_MPG.city" "New\_MPG.highway"

########################  
#중복없는 유일한(distinct, unique) 값 추출: distinct()  
#dictinct(dataframe, 기준var1, 기준var2, ...)  
  
names(Cars93)

## [1] "Manufacturer" "Model" "Type"   
## [4] "Min.Price" "Price" "Max.Price"   
## [7] "MPG.city" "MPG.highway" "AirBags"   
## [10] "DriveTrain" "Cylinders" "EngineSize"   
## [13] "Horsepower" "RPM" "Rev.per.mile"   
## [16] "Man.trans.avail" "Fuel.tank.capacity" "Passengers"   
## [19] "Length" "Wheelbase" "Width"   
## [22] "Turn.circle" "Rear.seat.room" "Luggage.room"   
## [25] "Weight" "Origin" "Make"

#Cars93 데이터 프레임에서 '차종(Type)'과 '생산국-미국여부(Origin)' 변수를 기준으로   
#중복없는 유일한 값을 추출  
Cars93 %>% distinct( Origin )

## Origin  
## 1 non-USA  
## 2 USA

Cars93 %>% distinct( Type )

## Type  
## 1 Small  
## 2 Midsize  
## 3 Compact  
## 4 Large  
## 5 Sporty  
## 6 Van

Cars93 %>% distinct( Origin, Type )

## Type Origin  
## 1 Small non-USA  
## 2 Midsize non-USA  
## 3 Compact non-USA  
## 4 Midsize USA  
## 5 Large USA  
## 6 Compact USA  
## 7 Sporty USA  
## 8 Van USA  
## 9 Small USA  
## 10 Sporty non-USA  
## 11 Van non-USA

unique(Cars93[, c("Origin", "Type")])

## Origin Type  
## 1 non-USA Small  
## 2 non-USA Midsize  
## 3 non-USA Compact  
## 6 USA Midsize  
## 7 USA Large  
## 12 USA Compact  
## 14 USA Sporty  
## 16 USA Van  
## 23 USA Small  
## 40 non-USA Sporty  
## 56 non-USA Van

# rows names가 dataframe의 row 번호로 반환되는 차이  
  
  
  
########################  
#무작위 표본 데이터 추출 : sample\_n(), sample\_frac()  
  
#sample\_n(dataframe, a fixed number) : 특정 개수만큼 무작위 추출  
#Cars93 데이터 프레임 (1~5 변수만 사용) 에서 10개의 관관측치를 무작위로 추출  
Cars93 %>% select(1:5) %>% sample\_n(10)

## Manufacturer Model Type Min.Price Price  
## 22 Chrysler Imperial Large 29.5 29.5  
## 81 Subaru Loyale Small 10.5 10.9  
## 51 Lincoln Continental Midsize 33.3 34.3  
## 60 Mercury Capri Sporty 13.3 14.1  
## 90 Volkswagen Passat Compact 17.6 20.0  
## 59 Mercedes-Benz 300E Midsize 43.8 61.9  
## 68 Oldsmobile Achieva Compact 13.0 13.5  
## 39 Geo Metro Small 6.7 8.4  
## 52 Lincoln Town\_Car Large 34.4 36.1  
## 1 Acura Integra Small 12.9 15.9

Cars93 %>% select(1:5) %>% sample\_n(10)

## Manufacturer Model Type Min.Price Price  
## 76 Pontiac Grand\_Prix Midsize 15.4 18.5  
## 33 Ford Tempo Compact 10.4 11.3  
## 42 Honda Civic Small 8.4 12.1  
## 38 Ford Crown\_Victoria Large 20.1 20.9  
## 56 Mazda MPV Van 16.6 19.1  
## 12 Chevrolet Cavalier Compact 8.5 13.4  
## 3 Audi 90 Compact 25.9 29.1  
## 66 Nissan Quest Van 16.7 19.1  
## 18 Chevrolet Caprice Large 18.0 18.8  
## 93 Volvo 850 Midsize 24.8 26.7

#sample\_frac(dataframe, a fixed fraction) : 특정 비율만큼 무작위 추출  
#Cars93 데이터 프레임 (1~5 변수만 사용) 에서 10%의 관측치를 무작위로 추출  
nrow(Cars93)

## [1] 93

nrow(Cars93)\*0.1

## [1] 9.3

Cars93 %>% select(1:5) %>% sample\_frac(0.1) #9개 추출

## Manufacturer Model Type Min.Price Price  
## 67 Nissan Maxima Midsize 21.0 21.5  
## 90 Volkswagen Passat Compact 17.6 20.0  
## 24 Dodge Shadow Small 8.4 11.3  
## 83 Suzuki Swift Small 7.3 8.6  
## 16 Chevrolet Lumina\_APV Van 14.7 16.3  
## 81 Subaru Loyale Small 10.5 10.9  
## 12 Chevrolet Cavalier Compact 8.5 13.4  
## 46 Hyundai Scoupe Sporty 9.1 10.0  
## 60 Mercury Capri Sporty 13.3 14.1

#smaple\_n(dataframe, n, replace = TRUE) : 복원 추출  
#위 두 함수는 비복원추출이었음(defalt 옵션), 복원추출을 하고 싶다면 replace=TRUE  
#Cars93 데이터 프레임(1~5번까지 변수만 사용)에서 20개의 관측치를 무작위 복원추출  
Cars93 %>% select(1:5) %>% sample\_n(20, replace=TRUE)

## Manufacturer Model Type Min.Price Price  
## 83 Suzuki Swift Small 7.3 8.6  
## 33 Ford Tempo Compact 10.4 11.3  
## 20 Chrylser Concorde Large 18.4 18.4  
## 52 Lincoln Town\_Car Large 34.4 36.1  
## 57 Mazda RX-7 Sporty 32.5 32.5  
## 70 Oldsmobile Silhouette Van 19.5 19.5  
## 9 Buick Riviera Midsize 26.3 26.3  
## 37 Ford Taurus Midsize 15.6 20.2  
## 44 Hyundai Excel Small 6.8 8.0  
## 47 Hyundai Sonata Midsize 12.4 13.9  
## 23 Dodge Colt Small 7.9 9.2  
## 4 Audi 100 Midsize 30.8 37.7  
## 69 Oldsmobile Cutlass\_Ciera Midsize 14.2 16.3  
## 92 Volvo 240 Compact 21.8 22.7  
## 51 Lincoln Continental Midsize 33.3 34.3  
## 22 Chrysler Imperial Large 29.5 29.5  
## 76 Pontiac Grand\_Prix Midsize 15.4 18.5  
## 84 Toyota Tercel Small 7.8 9.8  
## 50 Lexus SC300 Midsize 34.7 35.2  
## 90 Volkswagen Passat Compact 17.6 20.0

#dataframe %>% group\_by(factor\_var) %>% sample\_n(size) : 집단별 층화 표본  
#집단, 그룹별로 동일한 수의 표본을 무작위 추출해서 분석해야 하는 경우   
  
#Cars93 데이터 프레임에서 '제조국가\_미국여부(Origin)'의 'USA', 'non-USA' 요인 속성별로 각 10개씩의 표본  
Cars93 %>% select(Manufacturer, Model, Origin) %>%   
 group\_by(Origin) %>% sample\_n(10)

## # A tibble: 20 x 3  
## # Groups: Origin [2]  
## Manufacturer Model Origin   
## <fct> <fct> <fct>   
## 1 Buick Riviera USA   
## 2 Saturn SL USA   
## 3 Lincoln Continental USA   
## 4 Ford Tempo USA   
## 5 Chevrolet Lumina\_APV USA   
## 6 Pontiac LeMans USA   
## 7 Chevrolet Camaro USA   
## 8 Dodge Spirit USA   
## 9 Ford Taurus USA   
## 10 Buick Roadmaster USA   
## 11 Nissan Altima non-USA  
## 12 Hyundai Excel non-USA  
## 13 Mazda Protege non-USA  
## 14 Acura Legend non-USA  
## 15 Subaru Loyale non-USA  
## 16 Volvo 850 non-USA  
## 17 Toyota Previa non-USA  
## 18 Lexus SC300 non-USA  
## 19 Volkswagen Passat non-USA  
## 20 Volkswagen Fox non-USA

#####################  
#새로운 변수 생성 : nutate(), transmutate()  
  
#mutate(dataframe, 새로운변수= 기존변수 조합한 수식, ...): 기존변수+신규변수 모두 keep  
Cars93\_1 <- Cars93 %>% select( Model, Min.Price, Max.Price) %>% head(10) # subset for better printing  
Cars93\_1

## Model Min.Price Max.Price  
## 1 Integra 12.9 18.8  
## 2 Legend 29.2 38.7  
## 3 90 25.9 32.3  
## 4 100 30.8 44.6  
## 5 535i 23.7 36.2  
## 6 Century 14.2 17.3  
## 7 LeSabre 19.9 21.7  
## 8 Roadmaster 22.6 24.9  
## 9 Riviera 26.3 26.3  
## 10 DeVille 33.0 36.3

Cars93\_1 <- Cars93\_1 %>% mutate(Price\_range = Max.Price - Min.Price,   
 Price\_Min\_Max\_ratio = Max.Price / Min.Price)  
Cars93\_1

## Model Min.Price Max.Price Price\_range Price\_Min\_Max\_ratio  
## 1 Integra 12.9 18.8 5.9 1.457364  
## 2 Legend 29.2 38.7 9.5 1.325342  
## 3 90 25.9 32.3 6.4 1.247104  
## 4 100 30.8 44.6 13.8 1.448052  
## 5 535i 23.7 36.2 12.5 1.527426  
## 6 Century 14.2 17.3 3.1 1.218310  
## 7 LeSabre 19.9 21.7 1.8 1.090452  
## 8 Roadmaster 22.6 24.9 2.3 1.101770  
## 9 Riviera 26.3 26.3 0.0 1.000000  
## 10 DeVille 33.0 36.3 3.3 1.100000

#transmute(dataframe, 새로운 변수 = 기존 변수 조합한 수식, ...): 신규 변수만 keep  
Cars93\_1 <- Cars93 %>% select( Model, Min.Price, Max.Price) %>% head(10) # subset for better printing  
Cars93\_1

## Model Min.Price Max.Price  
## 1 Integra 12.9 18.8  
## 2 Legend 29.2 38.7  
## 3 90 25.9 32.3  
## 4 100 30.8 44.6  
## 5 535i 23.7 36.2  
## 6 Century 14.2 17.3  
## 7 LeSabre 19.9 21.7  
## 8 Roadmaster 22.6 24.9  
## 9 Riviera 26.3 26.3  
## 10 DeVille 33.0 36.3

Cars93\_2 <- Cars93\_1 %>% transmute( Price\_range = Max.Price - Min.Price,   
 Price\_Min\_Max\_ratio = Max.Price / Min.Price)  
Cars93\_2

## Price\_range Price\_Min\_Max\_ratio  
## 1 5.9 1.457364  
## 2 9.5 1.325342  
## 3 6.4 1.247104  
## 4 13.8 1.448052  
## 5 12.5 1.527426  
## 6 3.1 1.218310  
## 7 1.8 1.090452  
## 8 2.3 1.101770  
## 9 0.0 1.000000  
## 10 3.3 1.100000

#####################  
#값 요약: summarise()  
  
  
#summarise(dataframe, mean, sd, ...) : 수치형 값에 대한 요약 통계량 계산  
  
# - mean(x, na.rm = TRUE) : 평균, 결측값을 제외하고 계산하려면 na.rm = TRUE 추가  
# - median(x, na.rm = TRUE) : 중앙값  
# - sd(x, na.rm = TRUE) : 표준편차  
# - min(x, na.rm = TRUE) : 최소값  
# - max(x, na.rm = TRUE) : 최대값  
# - IQR(x, na.rm = TRUE) : 사분위수 (Inter Quartile Range = Q3 - Q1)  
# - sum(x, na.rm = TRUE) : 합, 결측값을 제외하고 계산하려면 na.rm = TRUE 추가  
  
  
#Cars93 데이터 프레임에서 가격(Price)의 (a) 평균, (b) 중앙값, (c) 표준편차, (d) 최소값, (e) 최대값, (f) 사분위수(IQR), (g) 합계  
  
Cars93 %>% summarise(Price\_mean = mean(Price, na.rm = TRUE), # mean of Price  
 Price\_median = median(Price, na.rm = TRUE), # median of Price  
 Price\_sd = sd(Price, na.rm = TRUE), # standard deviation of Price  
 Price\_min = min(Price, na.rm = T), # min of Price  
 Price\_max = max(Price, na.rm = T), # max of Price  
 Price\_IQR = IQR(Price), na.rm = T, # IQR of Price  
 Price\_sum = sum(Price, na.rm = TRUE)) # sum of Price

## Price\_mean Price\_median Price\_sd Price\_min Price\_max Price\_IQR na.rm  
## 1 19.50968 17.7 9.65943 7.4 61.9 11.1 TRUE  
## Price\_sum  
## 1 1814.4

#summarise(dataframe, n(), n\_distinct(x), first(x), last(x), nth(x, n)) : 개수 계산, 관측값 indexing  
#   
# - n() : 관측치 개수 계산, x 변수 입력하지 않음  
# - n\_distinct(x) : 중복없는 유일한 관측치 개수 계산, 기준이 되는 x변수 입력함  
# - first(x) : 기준이 되는 x변수의 첫번째 관측치  
# - last(x) : 기준이 되는 x변수의 마지막 관측치  
# - nth(x, n) : 기준이 되는x변수의 n번째 관측치  
  
  
#Cars93\_1 데이터 프레임에서 (a) 총 관측치의 개수, (b) 제조사(Manufacturer)의 개수(유일한 값), (c) 첫번째 관측치의 제조사 이름, (d) 마지막 관측치의 제조사 이름, (e) 5번째 관측치의 제조사 이름  
Cars93\_1 <- Cars93[c(1:10), c("Manufacturer", "Model", "Type")] # subset for better print  
Cars93\_1

## Manufacturer Model Type  
## 1 Acura Integra Small  
## 2 Acura Legend Midsize  
## 3 Audi 90 Compact  
## 4 Audi 100 Midsize  
## 5 BMW 535i Midsize  
## 6 Buick Century Midsize  
## 7 Buick LeSabre Large  
## 8 Buick Roadmaster Large  
## 9 Buick Riviera Midsize  
## 10 Cadillac DeVille Large

Cars93\_1 %>% summarise( tot\_cnt = n(), # counting the number of all observations  
 Manufacturer\_dist\_cnt = n\_distinct(Manufacturer), # distinct number of var   
 First\_obs = first(Manufacturer), # first observation   
 Last\_obs = last(Manufacturer), # last observation   
 Nth\_5th\_obs = nth(Manufacturer, 5)) # n'th observation

## tot\_cnt Manufacturer\_dist\_cnt First\_obs Last\_obs Nth\_5th\_obs  
## 1 10 5 Acura Cadillac BMW

#summarise(group\_by(dataframe, factor\_var), mean, sd, ...): 그룹별 요약 통계ㄹ  
#Cars93 데이터 프레임에서 '차종(Type)' 별로 구분해서   
#(a) 전체 관측치 개수, (b) (중복 없이 센) 제조사 개수, (c) 가격(Price)의 평균과 (d) 가격의 표준편차  
  
group\_df <- Cars93 %>% group\_by(Type)  
group\_df %>% summarise(tot\_conut = n(), # counting the number of cars  
 Manufacturer\_dist\_cnt = n\_distinct(Manufacturer), # distinct number of var  
 Price\_mean = mean(Price, na.rm = TRUE), # mean of Price  
 Price\_sd = sd(Price, na.rm = TRUE) # standard deviation of Price  
 )

## # A tibble: 6 x 5  
## Type tot\_conut Manufacturer\_dist\_cnt Price\_mean Price\_sd  
## <fct> <int> <int> <dbl> <dbl>  
## 1 Compact 16 15 18.2 6.69  
## 2 Large 11 10 24.3 6.34  
## 3 Midsize 22 20 27.2 12.3   
## 4 Small 21 16 10.2 1.95  
## 5 Sporty 14 12 19.4 7.97  
## 6 Van 9 8 19.1 1.88

#summarise\_each() : 다수의 변수에 동일한 summarise 함수 적용  
#Cars93 데이터 프레임의 (i) 가격(Price) 변수와 (ii) 고속도로연비(MPG.highway) 의 두개의 변수에 대해   
#(a) 평균(mean), (b) 중앙값(median), (c) 표준편차(standard deviation) 의 3개의 함수를 동시에 적용  
Cars93 %>% summarise\_each( funs(mean, median, sd), Price, MPG.highway)

## `summarise\_each()` is deprecated.  
## Use `summarise\_all()`, `summarise\_at()` or `summarise\_if()` instead.  
## To map `funs` over a selection of variables, use `summarise\_at()`

## Price\_mean MPG.highway\_mean Price\_median MPG.highway\_median Price\_sd  
## 1 19.50968 29.08602 17.7 28 9.65943  
## MPG.highway\_sd  
## 1 5.331726

###############################################  
#그룹별로 행의 개수 세기   
#dataframe %>% group\_by( factor ) %>% summarise(n=n())  
#차 종류(Type) 별로 그룹개수 세기  
Cars93 %>% group\_by(Type) %>% summarise(n=n())

## # A tibble: 6 x 2  
## Type n  
## <fct> <int>  
## 1 Compact 16  
## 2 Large 11  
## 3 Midsize 22  
## 4 Small 21  
## 5 Sporty 14  
## 6 Van 9

Cars93 %>% group\_by(Type) %>% summarise(n=n(), n\_distinct\_maker= n\_distinct(Manufacturer))

## # A tibble: 6 x 3  
## Type n n\_distinct\_maker  
## <fct> <int> <int>  
## 1 Compact 16 15  
## 2 Large 11 10  
## 3 Midsize 22 20  
## 4 Small 21 16  
## 5 Sporty 14 12  
## 6 Van 9 8

#dataframe %>% group\_by( factor ) %>% tally()  
Cars93 %>% group\_by(Type) %>% tally()

## # A tibble: 6 x 2  
## Type n  
## <fct> <int>  
## 1 Compact 16  
## 2 Large 11  
## 3 Midsize 22  
## 4 Small 21  
## 5 Sporty 14  
## 6 Van 9

#dataframe %>%count( factor )  
Cars93 %>% count(Type)

## # A tibble: 6 x 2  
## Type n  
## <fct> <int>  
## 1 Compact 16  
## 2 Large 11  
## 3 Midsize 22  
## 4 Small 21  
## 5 Sporty 14  
## 6 Van 9

# doing both grouping and counting (no need for group\_by())  
  
  
  
###############################################  
###############################################  
#join 함수  
####data frame을 조건에 맞게 join 시켜줌 ,sql 하고 비슷  
####left , right , inner, full join 이 있음  
set.seed(1)  
  
log <- data.frame( user\_id = sample(c(1,2,3), 10, TRUE), item\_id =  
 sample( c(1,2,3), 10, TRUE ), correct = sample(c(0,1), 10, TRUE) )  
users <- data.frame( user\_id = c(1,2,4), age = c(20,20,30) )  
items <- data.frame( item\_id = 1:3, item = c("1+1","2\*2","3/3") )  
  
log ; users; items

## user\_id item\_id correct  
## 1 1 1 1  
## 2 2 1 0  
## 3 2 3 1  
## 4 3 2 0  
## 5 1 3 0  
## 6 3 2 0  
## 7 3 3 0  
## 8 2 3 0  
## 9 2 2 1  
## 10 1 3 0

## user\_id age  
## 1 1 20  
## 2 2 20  
## 3 4 30

## item\_id item  
## 1 1 1+1  
## 2 2 2\*2  
## 3 3 3/3

#user\_id 가 key 값 으로 dataframe 이 join된다  
  
log %>% left\_join(users,"user\_id")

## user\_id item\_id correct age  
## 1 1 1 1 20  
## 2 2 1 0 20  
## 3 2 3 1 20  
## 4 3 2 0 NA  
## 5 1 3 0 20  
## 6 3 2 0 NA  
## 7 3 3 0 NA  
## 8 2 3 0 20  
## 9 2 2 1 20  
## 10 1 3 0 20

log %>% left\_join(users,"user\_id") %>% left\_join(items,"item\_id")

## user\_id item\_id correct age item  
## 1 1 1 1 20 1+1  
## 2 2 1 0 20 1+1  
## 3 2 3 1 20 3/3  
## 4 3 2 0 NA 2\*2  
## 5 1 3 0 20 3/3  
## 6 3 2 0 NA 2\*2  
## 7 3 3 0 NA 3/3  
## 8 2 3 0 20 3/3  
## 9 2 2 1 20 2\*2  
## 10 1 3 0 20 3/3

log %>% right\_join(users,"user\_id")

## user\_id item\_id correct age  
## 1 1 1 1 20  
## 2 1 3 0 20  
## 3 1 3 0 20  
## 4 2 1 0 20  
## 5 2 3 1 20  
## 6 2 3 0 20  
## 7 2 2 1 20  
## 8 4 NA NA 30

log %>% inner\_join(users,"user\_id")

## user\_id item\_id correct age  
## 1 1 1 1 20  
## 2 2 1 0 20  
## 3 2 3 1 20  
## 4 1 3 0 20  
## 5 2 3 0 20  
## 6 2 2 1 20  
## 7 1 3 0 20

log %>% inner\_join(users,"user\_id") %>% inner\_join(items,"item\_id")

## user\_id item\_id correct age item  
## 1 1 1 1 20 1+1  
## 2 2 1 0 20 1+1  
## 3 2 3 1 20 3/3  
## 4 1 3 0 20 3/3  
## 5 2 3 0 20 3/3  
## 6 2 2 1 20 2\*2  
## 7 1 3 0 20 3/3

log %>% full\_join(users,"user\_id")

## user\_id item\_id correct age  
## 1 1 1 1 20  
## 2 2 1 0 20  
## 3 2 3 1 20  
## 4 3 2 0 NA  
## 5 1 3 0 20  
## 6 3 2 0 NA  
## 7 3 3 0 NA  
## 8 2 3 0 20  
## 9 2 2 1 20  
## 10 1 3 0 20  
## 11 4 NA NA 30