4장 데이터 프레임(Data Frame)

R에서 대부분의 데이터는 데이터 프레임 형식으로 존재. 데이터 프레임은 R 객체 중에서 많이 접하게 되는 객체 중 하나.

1. 데이터 프레임 생성
2. 데이터 프레임의 구성요소 선택(서브셋팅)
3. 데이터 프레임과 기초 통계량
4. 데이터 프레임 병합

1. 데이터 프레임 생성:

data.frame() 사용

```
> d<- data.frame(c(1,2,3), c(6,7,8))
> d # 행번호=1,2,3, 열이름(변수명)=c..2..3, c.6..7..8
> colnames(d)<-c("x","y") #변수명 부여
> d # 행번호=1,2,3, 열이름(변수명)=x,y
> str(d) #str(): 데이터 구조, 변수의 개수, 변수 명($ 사용해서), 변수의 속성, 값
> attributes(d)
```

데이터 프레임의 각 변수는 서로 다른 데이터 타입을 사용 가능

```
> x <- data.frame(name=c('Kim', 'Lee', 'Cho'),
gender=c('Male','Female','Female'),age=c(19,20,21) )
> str(d)
> attributes(d)
> x$age #데이터 프레임 x 에 있는 변수 age의 내용을 보기
```

2. 데이터 프레임의 서브셋팅: 구성요소 선택

데이터 프레임 서브셋팅은 subset() 함수, **"[ ]" 그리고 "$"**를 사용

```
> x[x$age>19,] # 나이가 19살보다 많은 사람의 이름, 성별, 나이(원래순)
> x[x$age>19, "name"] # 나이가 19살보다 많은 사람의 이름
> x[x$age>19,c("age", "name")] # 나이가 19살보다 많은 사람의 나이, 이름
> x[["age"]]; x$age

> x[x$age>19, 1] # 나이가 19살보다 많은 사람의 이름(1번째 열="name")
> x[x$age>19,c(3,1)] # 나이가 19살보다 많은 사람의 나이, 이름(3,1번째 열="age", "name")
> x[[3]]; x[,3]
```

표 4.2 head()와 tail() 함수: 데이터의 앞부분과 뒷부분만 간단히 볼 수 있는 함수

```
> data() # R에 내장되어 있는 데이터셋.
> help(cars) #  자동차 속도(mph)와 제동거리(ft) ~ 파일창
> str(cars)
> head(cars); tail(cars)
> cars[cars$speed>20,] # 속도가 20mph 큰 경우 속도와 제동거리
> cars[cars$speed>20,"dist"] # 속도가 20mph 큰 경우 제동거리
> mean(cars[cars$speed>20,"dist"]); max(cars[cars$speed>20,"dist"]) #min, median
> cars[cars$dist <=10,] # 제동거리가 10ft 이하인 경우 속도와 제동거리
```

3. 데이터 프레임과 기초 통계량

summary(): 데이터 프레임의 기초 통계량을 계산하는 generic 함수(일반함수)

양적(수치형)변수 ~ 최소값(min), 제1사분위수(Q1), 중앙값(median), 평균(mean),
제3사분위수(Q3), 최대값(max).
질적변수 ~ 빈도표(frequency table).

```
> summary(cars)
> help(warpbreaks)
> summary(warpbreaks)
```

4. 데이터 프레임 병합

표 4.3 데이터 프레임 병합함수

```
> df1<-data.frame(name=c('KIM','LEE','CHO'), math=c(90,100,95),stringsAsFactors= F)
# 데이터 프레임 병합시 문자형 변수 있는 경우, stringsAsFactors= F.

> df2<-data.frame(name=c('KIM','YOON'), sci=c(100,95), stringsAsFactors= F)
> df1
> df2
> merge(df1, df2, by='name', all=T) # 두 데이터프레임 모든 행 합치기
> merge(df1, df2, by='name')  # 공통 이름에 대해서 합치기

# df1에 역사점수(hist)를 추가한 data.frame 만들기 - df3, df31
> df3<-data.frame(df1, hist=c(100,90,80))
> df31<-cbind(df1, hist=c(100,90,80))
> df3; df31
> str(df3); str(df31)
```

```
# df1에 행(Park, 85) 추가한  data.frame 만들기 - df41
> df41<-rbind(df1, c(name="Park",math=85)
> df41
> str(df41)

# 수학점수가 90 넘는 자료로
> df3[df3$math>90,]
# 수학점수가 90 넘는 사람의 이름
> df3[df3$math>90,"name"]
# 수학점수가 90 넘는 사람의 역사점수
> df3[df3$math>90,"hist"]
# 수학점수가 90 넘는 사람의 역사점수의 평균(or max, min, median)
> mean(df3[df3$math>90,"hist"])
```

5장 리스트(list)

벡터는 일차원 형태고 구성요소는 동일한 테이터 타입.
행렬은 동일한 데이터 타입의 벡터를 결합한 이차원 형태. 구성요소는 동일한 데이터 타입.
데이터 프레임은 행렬처럼 이차원 형태, 결합되는 벡터가 동일한 데이터 타입이 아니어도 됨.
리스트의 각 구성 요소는 벡터, 행렬, 데이터 프레임 그리고 리스트 중 하나가 될 수 있고.
각 구성 요소의 길이와 크기는 달라도 됨. 일차원 형태.

리스트 생성: list() 함수사용

```
> x<-list(1, c('a','b'), 1:3) # 3개의 구성요소
> str(x)
> x
> x[[3]] # 3번째 구성요소
> x[[3]][1] # 1번째 구성요소의 1번째 원소)
> x[[3]][2:3] # 1번째 구성요소의 2,3번째 원소

> mylist<- list(v=x, w=2:5, m=data.frame(x=c(1,2), y=c("F","M")))
# 3개의 구성요소, 구성요소 이름=v,w,m
> str(mylist)
> mylist
> mylist$v; mylist[1] # 1번째 구성요소(v)
> mylist$v[[1]] # 1번째 구성요소(v)의 1번째 원소(1)
> mylist$v[[2]] # 1번째 구성요소(v)의 2번째 원소('a','b')
> mylist$w[2:3] # w가 벡터인 경우, w의 2,3번째 원소
> mylist$m[1,]; mylist$m[,2]; # m가 행렬인 경우, m의 1번째 행, 2번째 열
```

```
> z<-vector(mode='list') # 빈 list
> z[['v']]<-list(c(1,2,3))
> z[['w']]<-2:5
> z[['m']]<-data.frame(x=c(1,2), y=c("F","M"))
> str(z)
> z
```

unlist(): 리스트의 구성 요소를 일차원으로 풀어서 벡터를 만드는 함수

```
> zv<-unlist(z)
> zv
```

리스트 합치기: c() 이용

```
> x1<-list(1,2,3); x2<-list(c('a','b','c'))
> x12<-c(x1,x2)
```