

2주차 R기초

데이터 프레임 & 리스트

교육부 운영진

| 목차

2.4 데이터프레임

- 2.4.1 데이터프레임 생성
- 2.4.2 데이터 인덱싱
- 2.4.3 데이터 개요 보기
- 2.4.4 데이터 부분 검색(filter)
- 2.4.5 데이터 값 변경
- 2.4.6 구조변경(변수 추가)
- 2.4.7 데이터 프레임 결합

2.5 리스트

- 2.5.1 리스트 생성
- 2.5.2 리스트 인덱싱
- 2.5.3 요소의 수정/삭제/추가

apply & aggregate

- 2.4.4.5 그룹 지어 보기
- 2.5.4 모든 요소에 일괄 반영하기

데이터 프레임 생성

2.4

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
3 #2.4.1
4 id <- c(1:4)
5 name <- c("갑", "을", "병", "정")
6 age <- c(22, 34, 30, 28)
7 isMarried <- c(F, T, F, T)
8 df <- data.frame(id, name, age, isMarried)
9 df
10
```

```
> df <- data.frame(id, name, age, isMarried)
> df
```

	id	name	age	isMarried
1	1	갑	22	FALSE
2	2	을	34	TRUE
3	3	병	30	FALSE
4	4	정	28	TRUE

1열

1행

* 데이터 프레임의 형태



데이터의 가장 기본적인 형태

- 가로가 행
- 세로가 열(변수)

변수명 지정

입력 값 전체가 변수명으로 지정 되어 있음.

```
10
11 data.frame(c(1:4),c("갑","을","병","정"),c(22,34,30,28),c(F,T,F,T))
12 data.frame(id=c(1:4),name=c("갑","을","병","정"),age=c(22,34,30,28),
13             isMarried=c(F,T,F,T))
14
```

```
> data.frame(c(1:4),c("갑","을","병","정"),c(22,34,30,28),c(F,T,F,T))
  c.1.4.  c..갑....을....병....정..  c.22..34..30..28.  c.F..T..F..T.
1      1      갑      22      FALSE
2      2      을      34      TRUE
3      3      병      30      FALSE
4      4      정      28      TRUE
> data.frame(id=c(1:4),name=c("갑","을","병","정"),age=c(22,34,30,28),
+             isMarried=c(F,T,F,T))
  id name age isMarried
1  1  갑  22   FALSE
2  2  을  34   TRUE
3  3  병  30   FALSE
4  4  정  28   TRUE
```

옵션: *stringsAsFactors*

str(데이터명/변수명)

- 데이터나 변수의 데이터 타입 or 기본적인 정보

```
15 str(df)
16 df <- data.frame(id,name,age,isMarried,stringsAsFactors = F)
17 str(df$name)
```

T: 문자열을 자동으로 팩터로 변환
F: 설정 해제

```
> str(df)
'data.frame':  4 obs. of  4 variables:
 $ id      : int  1 2 3 4
 $ name    : Factor w/ 4 levels "갑","병","을",...: 1 3 2 4
 $ age     : num  22 34 30 28
 $ isMarried: logi  FALSE TRUE FALSE TRUE
> df <- data.frame(id,name,age,isMarried,stringsAsFactors = F)
> str(df$name)
chr [1:4] "갑" "을" "병" "정"
```

데이터 인덱싱(특정 위치 값 반환)

2.4

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
19 #2.4.2 데이터 인덱싱
20 df
21 df[2,3] #[행, 열]
22 df[c(1,2),c(3,4)]
23
24 df[,c(3,4)] ①
25 df[c(1,2),]
26
27 df[,c("id","name")]
28
```

데이터 프레임 명[행, 열]

- 원하는 위치의 데이터 추출
- 행/열의 번호나 이름 입력

- ① [, 열번호] : 모든 행 추출
- [행번호 ,] : 모든 열 추출

```
> df[2,3] #[행, 열]
[1] 34
> df[c(1,2),c(3,4)]
  age isMarried
1  22      FALSE
2  34       TRUE
>
> df[,c(3,4)]
  age isMarried
1  22      FALSE
2  34       TRUE
3  30      FALSE
4  28       TRUE
```

```
> df[c(1,2),]
  id name age isMarried
1  1   갑  22      FALSE
2  2   을  34       TRUE
> df[,c("id","name")]
  id name
1  1   갑
2  2   을
3  3 병정
4  4   정
```

데이터 인덱싱(특정 위치 값 반환)

2.4

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
19 #2.4.2 데이터 인덱싱
20 df
21 df[2,3] #[행, 열]
22 df[c(1,2),c(3,4)]
23
24 df[,c(3,4)]
25 df[c(1,2),]
26
27 df[,c("id", "name")]
28
```

```
> df
  id name age isMarried
1  1  갑  22     FALSE
2  2  을  34      TRUE
3  3  병  30     FALSE
4  4  정  28      TRUE
```

```
> df[2,3] #[행, 열]
[1] 34
> df[c(1,2),c(3,4)]
  age isMarried
1  22     FALSE
2  34      TRUE
>
> df[,c(3,4)]
  age isMarried
1  22     FALSE
2  34      TRUE
3  30     FALSE
4  28      TRUE
```

```
> df[c(1,2),]
  id name age isMarried
1  1  갑  22     FALSE
2  2  을  34      TRUE
> df[,c("id", "name")]
  id name
1  1  갑
2  2  을
3  3  병
4  4  정
```

데이터 인덱싱(특정 변수에서)

2.4

```
29 df$name #벡터 형태로 출력
30 class(df$name)
31 df$name[2]
32 df$name[c(1,3)]
```

데이터명 \$ 변수명

- 데이터에서 특정 변수를 호출하고 싶다면 \$ 표시를 이용.

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
> df$name #벡터 형태로 출력
[1] "갑" "을" "병" "정"
> class(df$name)
[1] "character"
> df$name[2]
[1] "을"
> df$name[c(1,3)]
[1] "갑" "병"
```

벡터로 출력하기 때문에 벡터방식으로 인덱싱

데이터 개요 보기

2.4

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
34 #2.4.3 데이터 개요 보기
35 iris
36 str(iris)
37 str(iris$Sepal.Length)
```

iris 데이터

- R 기본 내장 데이터
- 붓꽃의 3가지 종(setosa, versicolor, virginica)에 대해 꽃받침(sepal)과 꽃잎(petal)의 길이를 정리한 데이터

```
> str(iris)
'data.frame': 150 obs. of 5 variables:
 $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ..
 $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1
 $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.
 $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.
 $ Species      : Factor w/ 3 levels "setosa","versicolor",...
 1 ...
> str(iris$Sepal.Length)
num [1:150] 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
```

1 150개 데이터
5개 변수

2 변수명

데이터 개요보기

2.4

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
40 head(iris)
41 head(iris,10)
42 tail(iris)
43 levels(iris$Species)
```

head(데이터명)

- 데이터의 상위 6개 행을 추출함

```
> head(iris)
  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
4          4.6         3.1          1.5          0.2  setosa
5          5.0         3.6          1.4          0.2  setosa
6          5.4         3.9          1.7          0.4  setosa
```

데이터 개요보기

2.4

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
40 head(iris)
41 head(iris,10)
42 tail(iris)
43 levels(iris$Species)
```

head(데이터명, n)

- 데이터의 상위 n개 행을 추출함
- 원하는 개수(n)을 입력
- n 입력 안할 시 자동으로 6개 추출

```
> head(iris,10)
  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
4          4.6         3.1          1.5          0.2  setosa
5          5.0         3.6          1.4          0.2  setosa
6          5.4         3.9          1.7          0.4  setosa
7          4.6         3.4          1.4          0.3  setosa
8          5.0         3.4          1.5          0.2  setosa
9          4.4         2.9          1.4          0.2  setosa
10         4.9         3.1          1.5          0.1  setosa
```

데이터 개요보기

2.4

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
40 head(iris)
41 head(iris,10)
42 tail(iris)
43 levels(iris$Species)
```

tail(데이터명)

- 데이터의 하위 6개 행을 추출함
- head와 마찬가지로 원하는 개수 입력할 수 있음.

```
> tail(iris)
      Sepal.Length Sepal.Width Petal.Length Petal.Width  Species
145           6.7         3.3         5.7         2.5 virginica
146           6.7         3.0         5.2         2.3 virginica
147           6.3         2.5         5.0         1.9 virginica
148           6.5         3.0         5.2         2.0 virginica
149           6.2         3.4         5.4         2.3 virginica
150           5.9         3.0         5.1         1.8 virginica
> levels(iris$Species)
[1] "setosa"      "versicolor" "virginica"
```

데이터 개수 세는 함수

2.4

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
45 #개수 세는 함수들
46 nrow(iris)
47 ncol(iris) #column 약자
48 dim(iris)
49 table(iris$Species)
50
```

```
> nrow(iris)
[1] 150
> ncol(iris) #column 약자
[1] 5
> dim(iris)
[1] 150 5
> table(iris$Species)
```

setosa	versicolor	virginica
50	50	50

150개의 행과 5개의 열로 구성

nrow(데이터명)

- 데이터의 열 개수 반환

ncol(데이터명)

- 데이터의 행 개수 반환

dim(데이터명)

- 데이터의 차원(행, 열 개수) 반환

데이터 기본 연산 함수

2.4

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
51 #데이터 기본 연산 함수
52 summary(iris)
53 min(iris$Sepal.Length)
54 max(iris$Sepal.Length)
55 median(iris$Sepal.Length)
56 mean(iris$Sepal.Length)
57 quantile(iris$Sepal.Length)
```

summary(데이터명)

- 각 변수에 대한 통계량 계산
- 최솟값, 사분위수, 최댓값

```
> summary(iris)
```

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
Min. :4.300	Min. :2.000	Min. :1.000	Min. :0.100	setosa :50
1st Qu.:5.100	1st Qu.:2.800	1st Qu.:1.600	1st Qu.:0.300	versicolor:50
Median :5.800	Median :3.000	Median :4.350	Median :1.300	virginica :50
Mean :5.843	Mean :3.057	Mean :3.758	Mean :1.199	
3rd Qu.:6.400	3rd Qu.:3.300	3rd Qu.:5.100	3rd Qu.:1.800	
Max. :7.900	Max. :4.400	Max. :6.900	Max. :2.500	

데이터 기본 연산 함수

2.4

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
51 #데이터 기본 연산 함수
52 summary(iris)
53 min(iris$Sepal.Length)
54 max(iris$Sepal.Length)
55 median(iris$Sepal.Length)
56 mean(iris$Sepal.Length)
57 quantile(iris$Sepal.Length)
```

```
> summary(iris)
Sepal.Length
Min.      :4.300
1st Qu.   :5.100
Median    :5.800
Mean      :5.843
3rd Qu.   :6.400
Max.      :7.900
```

min(데이터명) : 최솟값

max(데이터명) : 최댓값

median(데이터명) : 중앙값

mean(데이터명) : 평균

quantile(데이터명) : 사분위수

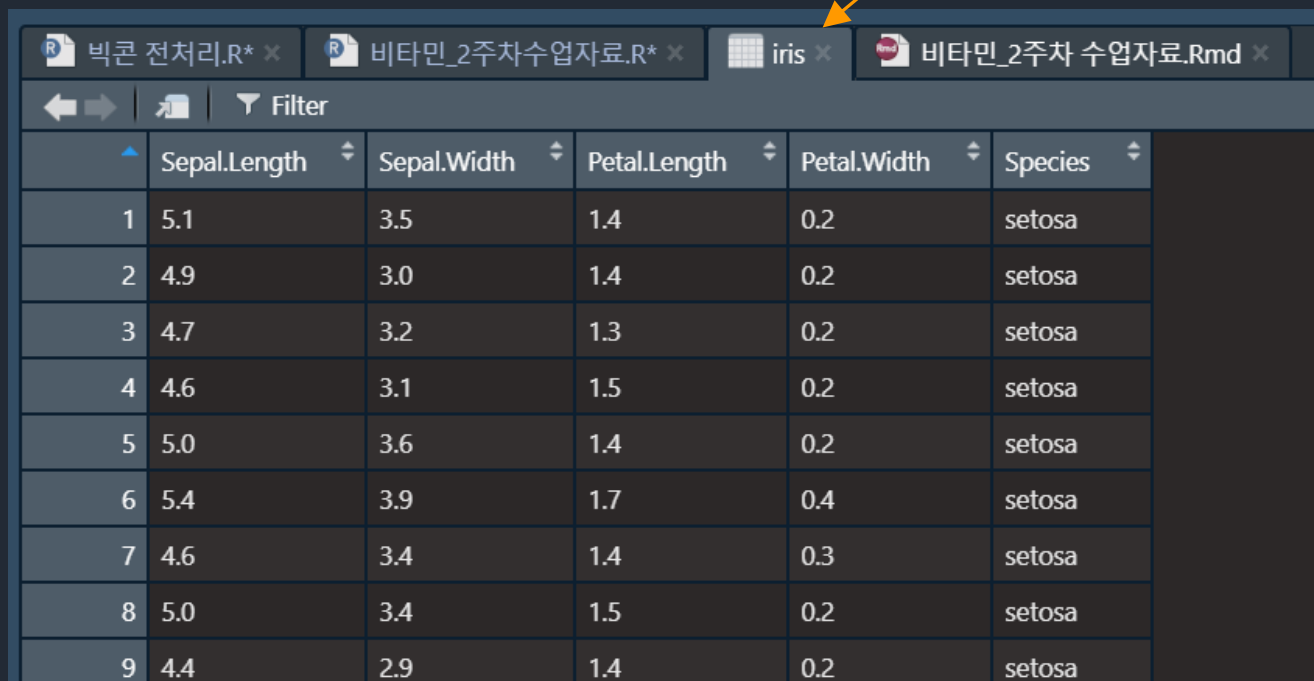
```
> min(iris$Sepal.Length)
[1] 4.3
> max(iris$Sepal.Length)
[1] 7.9
> median(iris$Sepal.Length)
[1] 5.8
> mean(iris$Sepal.Length)
[1] 5.843333
> quantile(iris$Sepal.Length)
 0%   25%   50%   75%  100%
4.3  5.1  5.8  6.4  7.9
```

데이터 탐색

```
59 #2.4.4 데이터 탐색
60 view(iris)
61 iris
```

View(데이터명)
- 새로운 탭으로 보여줌

데이터명
- 콘솔 창에서 보여줌



	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
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa

데이터 탐색

2.4

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
59 #2.4.4 데이터 탐색
60 view(iris)
61 iris
```

View(데이터명)

- 새로운 탭으로 보여줌

데이터명

- 콘솔 창에서 보여줌

		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
4		4.6	3.1	1.5	0.2	setosa
5		5.0	3.6	1.4	0.2	setosa
6		5.4	3.9	1.7	0.4	setosa
7		4.6	3.4	1.4	0.3	setosa
8		5.0	3.4	1.5	0.2	setosa
9		4.4	2.9	1.4	0.2	setosa
10		4.9	3.1	1.5	0.1	setosa
11		5.4	3.7	1.5	0.2	setosa
12		4.8	3.4	1.6	0.2	setosa

`subset()` 함수

`subset(데이터, 조건)`

- 조건에 부합하는 데이터만 추출
- '그리고' = & '또는' = |

```
63 subset(iris, Sepal.Length > 7)
64 subset(iris, Sepal.Length > 7 & Petal.Length < 6.5)
65 subset(iris, Sepal.Length > 7 | Petal.Length > 6.5)
66 subset(iris, Sepal.Length > max(Sepal.Length)-1)
67 subset(iris, Sepal.Length > 5.1 & Species=="setosa")
```

```
> subset(iris, Sepal.Length > 7)
  1 Sepal.Length Sepal.Width Petal.Length Petal.Width Species
103      7.1      3.0      5.9      2.1 virginica
106      7.6      3.0      6.6      2.1 virginica
108      7.3      2.9      6.3      1.8 virginica
110      7.2      3.6      6.1      2.5 virginica
118      7.7      3.8      6.7      2.2 virginica
119      7.7      2.6      6.9      2.3 virginica
123      7.7      2.8      6.7      2.0 virginica
126      7.2      3.2      6.0      1.8 virginica
130      7.2      3.0      5.8      1.6 virginica
131      7.4      2.8      6.1      1.9 virginica
132      7.9      3.8      6.4      2.0 virginica
136      7.7      3.0      6.1      2.3 virginica
```

- ① 꽃받침 길이가 모두 7 초과

`subset()` 함수

2.4

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
63 subset(iris, Sepal.Length > 7)
64 subset(iris, Sepal.Length > 7 & Petal.Length < 6.5)
65 subset(iris, Sepal.Length > 7 | Petal.Length > 6.5)
66 subset(iris, Sepal.Length > max(Sepal.Length)-1)
67 subset(iris, Sepal.Length > 5.1 & Species=="setosa")
```

```
> subset(iris, Sepal.Length > 7 & Petal.Length < 6.5)
  1 Sepal.Length Sepal.Width Petal.Length Petal.Width Species
103      7.1      3.0      5.9      2.1 virginica
108      7.3      2.9      6.3      1.8 virginica
110      7.2      3.6      6.1      2.5 virginica
126      7.2      3.2      6.0      1.8 virginica
130      7.2      3.0      5.8      1.6 virginica
131      7.4      2.8      6.1      1.9 virginica
132      7.9      3.8      6.4      2.0 virginica
136      7.7      3.0      6.1      2.3 virginica
```

- ① 꽃받침길이가 7초과
이고 꽃잎 길이가
6.5 미만인 것 추출

subset() 함수

2.4

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
63 subset(iris, Sepal.Length > 7)
64 subset(iris, Sepal.Length > 7 & Petal.Length < 6.5)
65 subset(iris, Sepal.Length > 7 | Petal.Length > 6.5)
66 subset(iris, Sepal.Length > max(Sepal.Length)-1)
67 subset(iris, Sepal.Length > 5.1 & Species=="setosa")
```

```
> subset(iris, Sepal.Length > 7 | Petal.Length > 6.5)
  1 Sepal.Length Sepal.Width Petal.Length Petal.Width Species
103         7.1         3.0         5.9         2.1 virginica
106         7.6         3.0         6.6         2.1 virginica
108         7.3         2.9         6.3         1.8 virginica
110         7.2         3.6         6.1         2.5 virginica
118         7.7         3.8         6.7         2.2 virginica
119         7.7         2.6         6.9         2.3 virginica
123         7.7         2.8         6.7         2.0 virginica
126         7.2         3.2         6.0         1.8 virginica
130         7.2         3.0         5.8         1.6 virginica
131         7.4         2.8         6.1         1.9 virginica
132         7.9         3.8         6.4         2.0 virginica
136         7.7         3.0         6.1         2.3 virginica
```

- ① 꽃받침길이가 7초과
이거나 꽃잎 길이가
6.5 초과인 것 추출

`subset()` 함수(특정 열 추출)

`subset(데이터, 조건, select=열)`

- 추출하고자 하는 열번호/이름 입력
- 열 이름 선호됨.

1 # 특정 열(변수) 추출 방법

69 `subset(iris, Sepal.Length > 7 & Petal.Length < 6.5, select = c(1, 5))` # 변수 위치로 추출

70 `subset(iris, Sepal.Length > 7 & Petal.Length < 6.5, select = c(Sepal.Length, Species))`

71

2 `subset(iris, Sepal.Length > 7 & Petal.Length < 6.5, select = c(5, 1))` # 변수 순서 지정

1

	Sepal.Length	Species
103	7.1	virginica
108	7.3	virginica
110	7.2	virginica
126	7.2	virginica
130	7.2	virginica
131	7.4	virginica
132	7.9	virginica
136	7.7	virginica

2

	Species	Sepal.Length
103	virginica	7.1
108	virginica	7.3
110	virginica	7.2
126	virginica	7.2
130	virginica	7.2
131	virginica	7.4
132	virginica	7.9
136	virginica	7.7

출력값은 변하지 않고, 변수 순서만 바뀜.

subset() 함수(특정 열 제거)

특정 열(변수) 제거 방법

- 1

```
sub_dat <- subset(iris, Sepal.Length > 7 & Petal.Length < 6.5, select = -c(2,4))
sub_dat
subset(iris, Sepal.Length > 7 & Petal.Length < 6.5, select = -c(Sepal.Width, Petal.Width))
```
- 2

```
subset(sub_dat, Sepal.Length > 7 & Petal.Length < 6.5, select = -c(2,4))
```

1	Sepal.Length	Petal.Length	Species
103	7.1	5.9	virginica
108	7.3	6.3	virginica
110	7.2	6.1	virginica
126	7.2	6.0	virginica
130	7.2	5.8	virginica
131	7.4	6.1	virginica
132	7.9	6.4	virginica
136	7.7	6.1	virginica

2	Sepal.Length	Species
103	7.1	virginica
108	7.3	virginica
110	7.2	virginica
126	7.2	virginica
130	7.2	virginica
131	7.4	virginica
132	7.9	virginica
136	7.7	virginica

-c() 했던 Sepal.Length 와 Species 제거됨.

열 위치 입력했을 때의 위험성

[] 로 원하는 값 추출

데이터명[원하는 조건, 열]

- 원하는 조건을 행 자리에 쓰고, 뽑고 싶은 열을 열 자리에 입력.

```
82 iris[iris$Sepal.Length > 7 & iris$Petal.Length <= 6.5]  
83 iris[iris$Sepal.Length > 7 & iris$Petal.Length <= 6.5,]  
84  
85 # subset(iris, Sepal.Length > 7 & Petal.Length <= 6.5)  
86  
87 iris[iris$Sepal.Length > 7 & iris$Petal.Length <= 6.5, ][, c(1, 5)]  
88
```

```
> iris[iris$Sepal.Length > 7 & iris$Petal.Length <= 6.5]  
Error in `[.data.frame'](iris, iris$Sepal.Length > 7 & iris$Petal.Length <= :  
  undefined columns selected  
> iris[iris$Sepal.Length > 7 & iris$Petal.Length <= 6.5,]  
   Sepal.Length Sepal.Width Petal.Length Petal.Width   Species  
103          7.1         3.0         5.9         2.1 virginica  
108          7.3         2.9         6.3         1.8 virginica  
110          7.2         3.6         6.1         2.5 virginica  
126          7.2         3.2         6.0         1.8 virginica  
130          7.2         3.0         5.8         1.6 virginica  
131          7.4         2.8         6.1         1.9 virginica  
132          7.9         3.8         6.4         2.0 virginica  
136          7.7         3.0         6.1         2.3 virginica
```

[] 로 원하는 값 추출

2.4

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
82 iris[iris$Sepal.Length > 7 & iris$Petal.Length <= 6.5]
83 iris[iris$Sepal.Length > 7 & iris$Petal.Length <= 6.5,]
84
85 # subset(iris, Sepal.Length > 7 & Petal.Length <= 6.5)
86
87 iris[iris$Sepal.Length > 7 & iris$Petal.Length <= 6.5,][,c(1,5)]
88
```

```
> iris[iris$Sepal.Length > 7 & iris$Petal.Length <= 6.5,][,c(1,5)]
```

	Sepal.Length	Species
103	7.1	virginica
108	7.3	virginica
110	7.2	virginica
126	7.2	virginica
130	7.2	virginica
131	7.4	virginica
132	7.9	virginica
136	7.7	virginica

[] 로 추출한 것은 데이터 프레임 형식이므로,
그에 맞는 인덱싱으로 원하는 열 추출

attach(), detach() 함수

2.4

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
#attach/detach
iris[Sepal.Length > 7 & Petal.Length <= 6.5,]
attach(iris)
iris[Sepal.Length > 7 & Petal.Length <= 6.5,]
detach(iris)
iris[Sepal.Length > 7 & Petal.Length <= 6.5,]
```

attach(데이터명)

- \$없이 변수만 써도 됨.

detach(데이터명)

- attach 설정 해제.

```
> iris[Sepal.Length > 7 & Petal.Length <= 6.5,]
Error in `[.data.frame`(iris, Sepal.Length > 7 & Petal.Length <= 6.5, :
  object 'Sepal.Length' not found
> attach(iris)
> iris[Sepal.Length > 7 & Petal.Length <= 6.5,]
   Sepal.Length Sepal.Width Petal.Length Petal.Width  Species
103           7.1         3.0          5.9         2.1 virginica
108           7.3         2.9          6.3         1.8 virginica
110           7.2         3.6          6.1         2.5 virginica
126           7.2         3.2          6.0         1.8 virginica
130           7.2         3.0          5.8         1.6 virginica
131           7.4         2.8          6.1         1.9 virginica
132           7.9         3.8          6.4         2.0 virginica
136           7.7         3.0          6.1         2.3 virginica
> detach(iris)
> iris[Sepal.Length > 7 & Petal.Length <= 6.5,]
Error in `[.data.frame`(iris, Sepal.Length > 7 & Petal.Length <= 6.5, :
  object 'Sepal.Length' not found
```

[] 로 원하는 값 추출(drop=F 옵션)

2.4

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
96 # 변수 한 개만 인덱싱 하는 경우
97 head(iris[,c("Sepal.Length")])
98 head(iris[,c("Sepal.Length"),drop=F])
99
100 str(iris[,c("Sepal.Length")])
101 str(iris[,c("Sepal.Length"),drop=F])
```

변수 한 개만 선택하면 벡터로 출력

[행, 열, drop=F]

- 원래 차원 그대로 유지하여 출력

```
> head(iris[,c("Sepal.Length")])
[1] 5.1 4.9 4.7 4.6 5.0 5.4
> head(iris[,c("Sepal.Length"),drop=F])
  Sepal.Length
1           5.1
2           4.9
3           4.7
4           4.6
5           5.0
6           5.4
>
> str(iris[,c("Sepal.Length")])
num [1:150] 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
> str(iris[,c("Sepal.Length"),drop=F])
'data.frame':   150 obs. of  1 variable:
 $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4
```

which() 함수(*arr.ind=T*)

2.4

```
104 df
105 which(df==T)
106 which(df==T,arr.ind = T)
107
```

which(조건, arr.ind=T)

- which의 초기값은 벡터의 위치를 반환. 데이터 프레임의 경우 일렬로 나열하여 벡터로 만든 뒤 위치 반환하게 됨.

- arr.ind=T 를 할 경우 행,열 위치로 출력

2.5

함수

1 2 3 4 갑 을 병 정 22 34 ...

이렇게 나열하고 값 반환

```
> df
  id name age isMarried
1  1  갑   22     FALSE
2  2  을   34      TRUE
3  3  병   30     FALSE
4  4  정   28      TRUE
> which(df==T)
[1] 1 14 16
> which(df==T,arr.ind = T)
      row col
[1,]    1   1
[2,]    2   4
[3,]    4   4
```

정렬

```
ex_df <- data.frame(name=c("고광민", "김영석", "이정민", "최지은"),  
                    age=c(23, 24, 24, 23), sex=c("남", "남", "남", "여"),  
                    stringsAsFactors=F)
```

```
ex_df
```

```
order(ex_df$age)  
ex_df[order(ex_df$age),]
```

```
> ex_df  
  name age sex  
1 고광민 23 남  
2 김영석 24 남  
3 이정민 24 남  
4 최지은 23 여
```

```
> order(ex_df$age)  
[1] 1 4 2 3
```

```
> ex_df[order(ex_df$age),]  
  name age sex  
1 고광민 23 남  
4 최지은 23 여  
2 김영석 24 남  
3 이정민 24 남
```

order[정렬 하고자 하는 변수]

- 원하는 변수를 오름차순으로 정렬
- 단! 위치 값을 출력

→ 나이가 적은 순으로 위치값 출력

정렬(*decreasing=T* 옵션)

order[정렬하려는 변수, *decreasing=T*]

- 내림차순 정렬을 원할 때(초기값은 F)

```
ex_df[order(ex_df$age,decreasing = T),]  
ex_df[order(ex_df$age,ex_df$sex,decreasing = T),]
```

```
idx <- order(ex_df$age,ex_df$sex,decreasing = T)  
ex_df[idx,]
```

```
> ex_df[order(ex_df$age,decreasing = T),]
```

	name	age	sex
2	김영석	24	남
3	이정민	24	남
1	고광민	23	남
4	최지은	23	여

```
> ex_df[order(ex_df$age,ex_df$sex,decreasing = T),]
```

	name	age	sex
2	김영석	24	남
3	이정민	24	남
4	최지은	23	여
1	고광민	23	남

문자열의 경우 알파벳은 abcd...
한글은 가 나 다 라... 순으로 정렬

age가 같은 경우에는 sex
를 내림차순 정렬

데이터 값 변경

2.4

```
ex_df[1,3] <- "여"  
ex_df
```

2.4.1 `wh_idx <- which(ex_df=="여",arr.ind=T)`

2.4.2 `wh_idx`

2.4.3 `ex_df[wh_idx] <- "남"`

2.4.4 `ex_df`

2.4.5 데이터명[바꾸려는 행,열] <- 원하는 값

2.4.6

2.4.7

2.5

함수

`ex_df[wh_idx,]`이 아닌 이유
- 이미 `wh_idx`가 행,열의 형태로 저장 되어 있기 때문에

ex_df의 1번째 행 3번째 열의 값을 바꾸겠다!

```
> ex_df[1,3] <- "여"  
> ex_df
```

	name	age	sex
1	고광민	23	여
2	김영석	24	남
3	이정민	24	남
4	최지은	23	여

ex_df에서 "여"가 있는 위치 찾기

```
> wh_idx <- which(ex_df=="여",arr.ind=T)  
> wh_idx
```

	row	col
[1,]	1	3
[2,]	4	3

```
> ex_df[wh_idx] <- "남"  
> ex_df
```

	name	age	sex
1	고광민	23	남
2	김영석	24	남
3	이정민	24	남
4	최지은	23	남

데이터 값 변경

꽃받침의 길이가 중앙값
이상이면 1번째 변수를
over median으로 바꾸자!

```
attach(iris)
tail(iris[Sepal.Length>=median(Sepal.Length),])
iris[Sepal.Length>=median(Sepal.Length),1] <- "over median"
tail(iris[Sepal.Length>=median(Sepal.Length),])
detach(iris)
```

```
> tail(iris[Sepal.Length>=median(Sepal.Length),])
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
145          6.7         3.3          5.7         2.5 virginica
146          6.7         3.0          5.2         2.3 virginica
147          6.3         2.5          5.0         1.9 virginica
148          6.5         3.0          5.2         2.0 virginica
149          6.2         3.4          5.4         2.3 virginica
150          5.9         3.0          5.1         1.8 virginica
```

```
> iris[Sepal.Length>=median(Sepal.Length),1] <- "over median"
> tail(iris[Sepal.Length>=median(Sepal.Length),])
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
145 over median         3.3          5.7         2.5 virginica
146 over median         3.0          5.2         2.3 virginica
147 over median         2.5          5.0         1.9 virginica
148 over median         3.0          5.2         2.0 virginica
149 over median         3.4          5.4         2.3 virginica
150 over median         3.0          5.1         1.8 virginica
```

새로운 열(변수) 추가하기

2.4

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
data(iris)
```

```
iris_new <- iris
```

```
iris_new$new <- "최지은"
```

```
head(iris_new, 3)
```

```
head(iris, 3)
```

```
> head(iris_new, 3)
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species	new
1	5.1	3.5	1.4	0.2	setosa	최지은
2	4.9	3.0	1.4	0.2	setosa	최지은
3	4.7	3.2	1.3	0.2	setosa	최지은

```
> 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

```
>
```

data(내장데이터)

- R 내장 데이터가 수정 되었을 때 다시 초기 데이터로 불러오기.

데이터명\$추가하려는 변수명

- 새로운 변수 추가하고 싶을 때
- 가장 오른쪽에 추가됨.

새로운 열(변수) 제거하기

데이터명[, - 제거하려는 열번호/이름]

- 열 번호 보다는 변수 이름 넣는 것을 선호

```
iris_new <- iris_new[,-1]
#iris_new$Sepal.Length <- NULL 다른 방법 같은 결과
head(iris_new,3)
```

열 이름으로 제거하고 싶을 때

```
iris_new <- iris_new[,-1]
head(iris_new,3)
```

```
> iris_new <- iris_new[,-1]
> #iris_new$Sepal.Length <- NULL 다른 방법 같은 결과
> head(iris_new,3)
  Sepal.Width Petal.Length Petal.Width Species new
1          3.5          1.4          0.2  setosa 최지은
2          3.0          1.4          0.2  setosa 최지은
3          3.2          1.3          0.2  setosa 최지은
```

```
> iris_new <- iris_new[,-1]
> head(iris_new,3)
  Petal.Length Petal.Width Species new
1          1.4          0.2  setosa 최지은
2          1.4          0.2  setosa 최지은
3          1.3          0.2  setosa 최지은
```

열번호로 제거의 위험성

- 제거하는 코드를 여러 번 돌렸을 때, 의도하지 않은 새로운 변수들이 계속 지워지게 됨.

열(변수) 이름 바꾸기

2.4

```
iris_new <- iris
colnames(iris_new)
colnames(iris_new) <- c("a","b","c","d","e")
colnames(iris_new)
head(iris_new,3)
```

```
colnames(iris_new)[3] <- "3rd"
head(iris_new,3)
```

colnames(데이터명)

- 데이터의 모든 변수 이름 출력(벡터)

colnames(데이터명) <- 새로운 변수명

- 특정 변수명만 바꾸고 싶다면 인덱싱을 통해 바꿀 수 있음.

```
> iris_new <- iris
> colnames(iris_new)
[1] "Sepal.Length" "Sepal.Width"  "Petal.Length"
    "Petal.Width"  "Species"
> colnames(iris_new) <- c("a","b","c","d","e")
> colnames(iris_new)
[1] "a" "b" "c" "d" "e"
> head(iris_new,3)
      a    b    c    d    e
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
>
> colnames(iris_new)[3] <- "3rd"
> head(iris_new,3)
```

3번째 변수명을 3rd로 바꾸자!

```
      a    b 3rd    d    e
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
```

데이터 타입 변경하기

2.4

```
170 df <- data.frame(id,name,age,isMarried)
171 str(df)
172 df$name <- as.character(df$name)
173 str(df$name)
```

as.데이터 타입(변경하는 값)

- 원하는 데이터 타입으로 변경됨.

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
> str(df)
'data.frame':  4 obs. of  4 variables:
 $ id      : int  1 2 3 4
 $ name     : Factor w/ 4 levels "갑","병","을",...: 1 3 2 4
 $ age      : num  22 34 30 28
 $ isMarried: logi  FALSE TRUE FALSE TRUE
> df$name <- as.character(df$name)
> str(df$name)
chr [1:4] "갑" "을" "병" "정"
```

데이터 타입 변경하기

2.4

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
#데이터 타입 변경이 필요한 이유
sum(df$age)
df$age <- as.character(df$age)
sum(df$age)
```

R은 숫자형이 아닐 경우에는 연산을
하지 못함.

따라서 연산을 하려는 경우 꼭 숫자형
으로 변환을 해주어야 함.

```
> sum(df$age)
[1] 114
> df$age <- as.character(df$age)
> sum(df$age)
Error in sum(df$age) : invalid 'type' (character) of argument
~
```

데이터 프레임끼리 결합하기

2.4

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
ex_df <- data.frame(name=c("이정민", "김영석", "고광민", "최지은"),
                    age=c(24, 24, 23, 23), sex=c("남", "남", "남", "여"))
ex_df2 <- data.frame(name=c("김영석", "최지은", "고광민", "이정민"),
                    birth=c("0617", "0921", "1013", "0216"))

cbind(ex_df, ex_df2)

merge(ex_df, ex_df2, by=c("name"))
```

이름을 기준으로 붙이겠다.

```
> cbind(ex_df, ex_df2)
  name age sex  name birth
1 이정민 24 남 김영석 0617
2 김영석 24 남 최지은 0921
3 고광민 23 남 고광민 1013
4 최지은 23 여 이정민 0216

>
> merge(ex_df, ex_df2, by=c("name"))
  name age sex birth
1 고광민 23 남 1013
2 김영석 24 남 0617
3 이정민 24 남 0216
4 최지은 23 여 0921
```

cbind(df1, df2...)

- 데이터 프레임들을 결합할 때
- 오른쪽 끝에 붙여 나감

merge(df1, df2..., by=합치는 기준)

- cbind와 같이 오른쪽 끝에 붙여 나가지만 특정 열을 기준으로 정렬하여 붙임.

데이터 프레임끼리 결합하기

2.4

2.4.1

2.4.2

2.4.3

2.4.4

2.4.5

2.4.6

2.4.7

2.5

함수

```
rbind(ex_df, c("주은혁", 24, "남"))  
rbind(ex_df, c("주은혁", 24, "남", "0903"))
```

rbind(df1, df2...)

- 행 기준으로 결합
- 가장 아래쪽에 붙음
- 열 개수가 안 맞으면 자동으로 자름

```
> rbind(ex_df, c("주은혁", 24, "남"))  
  name age sex  
1 이정민 24 남  
2 김영석 24 남  
3 고광민 23 남  
4 최지은 23 여  
5 주은혁 24 남  
> rbind(ex_df, c("주은혁", 24, "남", "0903"))  
  name age sex  
1 이정민 24 남  
2 김영석 24 남  
3 고광민 23 남  
4 최지은 23 여  
5 주은혁 24 남
```

0903은 열개수가 안 맞아서 자동으로 제거됨.

리스트 생성

```
vec_1 <- c(1:5)
vec_2 <- rep(c(T,F),c(2,3))
vec_3 <- data.frame(name=c("a","b","c","d"),age=seq(22,28,2))
vec_list <- list(vec_1,vec_2,vec_3)
vec_list
```

```
> vec_list
```

```
[[1]]
```

```
[1] 1 2 3 4 5
```

vec_1 해당

```
[[2]]
```

```
[1] TRUE TRUE FALSE FALSE FALSE
```

```
[[3]]
```

	name	age
1	a	22
2	b	24
3	c	26
4	d	28

list(원하는 모든 데이터)

- list는 데이터의 구조에 영향을 받지 않음

요소

- 리스트에서의 변수

- 이름 지정하지 않으면 [[1]], [[2]]...

순서로 자동 생성

리스트 요소 이름 변경

```
vec_list <- list(v1=vec_1,v2=vec_2,v3=vec_3)
vec_list
```

```
names(vec_list)
names(vec_list) <- c("1st","2nd","3rd")
names(vec_list)
```

names(리스트명)

- 리스트의 요소 이름 모두 출력
- 요소 이름 변경 방법은 데이터 프레임에서의 변수명 변경 방법과 동일함.

```
> vec_list
$v1
[1] 1 2 3 4 5

$v2
[1] TRUE TRUE FALSE FALSE FALSE

$v3
  name age
1    a  22
2    b  24
3    c  26
4    d  28

>
> names(vec_list)
[1] "v1" "v2" "v3"
> names(vec_list) <- c("1st","2nd","3rd")
> names(vec_list)
[1] "1st" "2nd" "3rd"
```


데이터 타입 변경하기

2.4

2.5

2.5.1

2.5.2

2.5.3

함수

```
208 vec_list$`1st`  
209 vec_list[1]  
210 vec_list[[1]]  
211 class(vec_list[1])  
212 class(vec_list[[1]])  
213  
214 vec_list[[1]][1:3]  
215
```

리스트명[위치값]

- 리스트로 반환

리스트명[[위치값]]

- 벡터로 반환

```
> vec_list$`1st`  
[1] 1 2 3 4 5  
1 > vec_list[1]  
$`1st`  
[1] 1 2 3 4 5  
2 > vec_list[[1]]  
[1] 1 2 3 4 5  
> class(vec_list[1])  
[1] "list"  
> class(vec_list[[1]])  
[1] "integer"  
>  
> vec_list[[1]][1:3]  
[1] 1 2 3
```

1 vec_list에 있는 첫번째 리스트를 찾겠다!

2 vec_list의 첫번째 리스트가 가지고 있는 값에 접근하겠다!

리스트 요소 수정하기

2.4

2.5

2.5.1

2.5.2

2.5.3

함수

```
str(vec_list)
```

```
vec_list[[3]] <- c(2:6)  
vec_list[[3]]
```

```
vec_list[[3]][2] <- 99  
vec_list[[3]]
```

리스트명[[위치]] <- 새로운 값

- 반드시 겹대괄호를 써줘야 함

3번째 요소의 2번째 값을 99로 바꾸겠다!

- 겹대괄호를 쓰면 벡터로 출력되기 때문에
벡터 인덱싱 활용하면 됨.

```
> str(vec_list)  
List of 3  
 $ 1st: int [1:5] 1 2 3 4 5  
 $ 2nd: logi [1:5] TRUE TRUE FALSE FALSE FAL  
 $ 3rd: 'data.frame': 4 obs. of 2 variabl  
 ..$ name: Factor w/ 4 levels "a","b","c",'  
 ..$ age : num [1:4] 22 24 26 28  
  
>  
> vec_list[[3]] <- c(2:6)  
> vec_list[[3]]  
[1] 2 3 4 5 6  
>  
> vec_list[[3]][2] <- 99  
> vec_list[[3]]  
[1] 2 99 4 5 6
```

리스트 요소 수정하기

2.4

2.5

2.5.1

2.5.2

2.5.3

함수

```
vec_list$`3rd` <- c(10:15)
vec_list$`3rd`
vec_list$`3rd`[2] <- 99
vec_list$`3rd`
|
```

리스트명\$요소명 <- 새로운 값

- 요소명에 숫자가 있다면 `요소명`으로 입력해 주어야 함.
- 별도의 처리가 없어도 벡터로 출력

```
> vec_list$`3rd` <- c(10:15)
> vec_list$`3rd`
[1] 10 11 12 13 14 15
> vec_list$`3rd`[2] <- 99
> vec_list$`3rd`
[1] 10 99 12 13 14 15
~
```

3rd 라는 요소명에 숫자가 들어있기 때문에 `3rd`로 입력해 주어야 함.

리스트 요소 제거하기

```
vec_list$`3rd` <- NULL  
vec_list
```

```
vec_rm <- vec_list[[-2]]  
vec_rm  
class(vec_rm)
```

```
vec_rm2 <- vec_list[-2]  
vec_rm2  
class(vec_rm2)
```

리스트명 \$ 요소명 <- NULL

- 입력한 요소를 제거

리스트명[[-위치값]]

- 입력한 위치의 벡터값을 제거

리스트명[-위치값]

- 입력한 위치의 요소를 제거

```
> vec_list$`3rd` <- NULL  
> vec_list  
$`1st`  
[1] 1 2 3 4 5  
  
$`2nd`  
[1] TRUE TRUE FALSE FALSE FALSE
```

```
> vec_rm <- vec_list[[-2]]  
> vec_rm  
[1] 1 2 3 4 5  
> class(vec_rm)   벡터에서 제거했기 때문에  
[1] "integer"      벡터로 남음  
>  
> vec_rm2 <- vec_list[-2]  
> vec_rm2  
$`1st`  
[1] 1 2 3 4 5  
  
> class(vec_rm2)  
[1] "list"
```

리스트 요소 추가하기

2.4

```
vec_list$`3rd` <- c(T,F,F,T)
```

2.5

```
vec_list
```

```
vec_list[["4th"]] <- c("new vector")
```

```
vec_list
```

```
vec_list[[5]] <- c("5th vector")
```

```
vec_list
```

2.5.1

2.5.2

2.5.3

함수

리스트명 \$ 요소명 <- 새로운 값

리스트명[[요소명]] <- 새로운 값

리스트명[[요소위치]] <- 새로운 값

```
> vec_list$`3rd` <- c(T,F,F,T)
```

```
> vec_list
```

```
$`1st`
```

```
[1] 1 2 3 4 5
```

```
$`2nd`
```

```
[1] TRUE TRUE FALSE FALSE FALSE
```

```
$`3rd`
```

```
[1] TRUE FALSE FALSE TRUE
```

리스트 요소 추가하기

2.4

```
vec_list$`3rd` <- c(T,F,F,T)
vec_list
```

2.5

```
vec_list[["4th"]] <- c("new vector")
vec_list
vec_list[[5]] <- c("5th vector")
vec_list
```

2.5.1

2.5.2

2.5.3

함수

리스트명 \$ 요소명 <- 새로운 값

리스트명[[요소명]] <- 새로운 값

리스트명[[요소위치]] <- 새로운 값

```
> vec_list[["4th"]] <- c("new vector")
> vec_list
$`1st`
[1] 1 2 3 4 5

$`2nd`
[1] TRUE TRUE FALSE FALSE FALSE

$`3rd`
[1] TRUE FALSE FALSE TRUE

$`4th`
[1] "new vector"
```

리스트 요소 추가하기

2.4

```
vec_list$`3rd` <- c(T,F,F,T)
vec_list
vec_list[["4th"]] <- c("new vector")
vec_list
vec_list[[5]] <- c("5th vector")
vec_list
```

2.5

2.5.1

2.5.2

2.5.3

함수

리스트명 \$ 요소명 <- 새로운 값

리스트명[[요소명]] <- 새로운 값

리스트명[[요소위치]] <- 새로운 값

```
> vec_list[[5]] <- c("5th vector")
> vec_list
$`1st`
[1] 1 2 3 4 5

$`2nd`
[1] TRUE TRUE FALSE FALSE FALSE

$`3rd`
[1] TRUE FALSE FALSE TRUE

$`4th`
[1] "new vector"

[[5]]
[1] "5th vector"
```

`apply()` 함수

```
str(iris)
iris_setosa <- subset(iris, Species=="setosa",
                      select = c(-Species))

apply(iris_setosa, 1, mean) #행 별
apply(iris_setosa, 2, mean) #열 별
```

apply(데이터명, 1 or 2, 함수)

- 1은 행 별, 2는 열 별
- 데이터를 행/열 별로 함수를 적용시킴.

```
> apply(iris_setosa, 1, mean) #행 별
 1      2      3      4      5      6      7      8      9     10     11     12
2.550 2.375 2.350 2.350 2.550 2.850 2.425 2.525 2.225 2.400 2.700 2.500
 13     14     15     16     17     18     19     20     21     22     23     24
2.325 2.125 2.800 3.000 2.750 2.575 2.875 2.675 2.675 2.675 2.350 2.650
 25     26     27     28     29     30     31     32     33     34     35     36
2.575 2.450 2.600 2.600 2.550 2.425 2.425 2.675 2.725 2.825 2.425 2.400
 37     38     39     40     41     42     43     44     45     46     47     48
2.625 2.500 2.225 2.550 2.525 2.100 2.275 2.675 2.800 2.375 2.675 2.350
 49     50
2.675 2.475
> apply(iris_setosa, 2, mean)
Sepal.Length Sepal.Width Petal.Length Petal.Width
      5.006      3.428      1.462      0.246
```

행 별로 4개의 변수
값을 평균 낸 것

열 별로 모든 꽃의
평균을 낸 것

lapply() & *sapply()*

2.4

2.5

함수

```
korea_temp <- list("경기" = c(-10, 2, 1, -2),  
                  "강원" = c(0, -4, -5, -10))  
korea_temp #리스트 생성  
  
result_lapply <- lapply(korea_temp, mean)  
result_sapply <- sapply(korea_temp, mean)  
result_lapply; result_sapply
```

lapply(데이터, 함수)

sapply(데이터, 함수)

- 공통점: 리스트는 요소별로, 데이터 프레임은 변수별로 함수값을 계산해줌
- 차이점: lapply는 리스트 값, sapply는 벡터 값으로 출력.

```
> korea_temp #리스트 생성  
$경기  
[1] -10    2    1   -2  
  
$강원  
[1]    0   -4   -5  -10
```

```
> result_lapply; result_sapply  
$경기  
[1] -2.25  
  
$강원  
[1] -4.75
```

```
경기 강원  
-2.25 -4.75
```

lapply() & *sapply()*

```
lapply(iris[, -5], mean)
sapply(iris[, -5], mean)
```

iris는 데이터 프레임 형식
-> 변수에 따라서 평균 자동
계산

```
> lapply(iris[, -5], mean)
$Sepal.Length
[1] 5.843333

$Sepal.Width
[1] 3.057333

$Petal.Length
[1] 3.758

$Petal.Width
[1] 1.199333

> sapply(iris[, -5], mean)
Sepal.Length Sepal.Width Petal.Length Petal.Width
    5.843333    3.057333    3.758000    1.199333
```

입력되는 데이터의 형태에 상관없이 출력형태는 **lapply**는
리스트 **sapply**는 **벡터**로 항상 같음

tapply() & *aggregate()*

붓꽃 종에 따른 꽃받침
길이의 평균을 계산

```
tapply_dat <- tapply(iris$Sepal.Length, iris$Species, mean)
tapply_dat
```

```
aggre_dat <- aggregate(Sepal.Length~Species, iris, mean)
aggre_dat
class(tapply_dat); class(aggre_dat)
aggregate(.~Species, iris, mean)
```

tapply(목표변수, 그룹변수, 함수)

aggregate(목표변수~그룹변수, 데이터, 함수)

- 공통점: 입력한 그룹변수에 따른 목표변수의

함수값을 계산해 줌

- 차이점: *tapply*는 array, *aggregate*는 데이터

프레임 값으로 출력.

- ~ 표시

종속변수~독립변수

종속변수는 알고자 하는 값

- . 표시

모든 변수를 사용한다.

tapply() & *aggregate()*

```
> tapply_dat <- tapply(iris$Sepal.Length, iris$Species, mean)
```

```
> tapply_dat
```

	setosa	versicolor	virginica
	5.006	5.936	6.588

```
>
```

```
> aggre_dat <- aggregate(Sepal.Length~Species, iris, mean)
```

```
> aggre_dat
```

	Species	Sepal.Length
1	setosa	5.006
2	versicolor	5.936
3	virginica	6.588

```
> class(tapply_dat); class(aggre_dat)
```

```
[1] "array"
```

```
[1] "data.frame"
```

붓꽃의 종에 따라서 모든 변수들의 평균을 계산

```
> aggregate(.~Species, iris, mean)
```

	Species	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
1	setosa	5.006	3.428	1.462	0.246
2	versicolor	5.936	2.770	4.260	1.326
3	virginica	6.588	2.974	5.552	2.026

2주차 R기초

데이터 프레임 & 리스트

다음 이 시간에...