R과 Knitr 개발 실무 – 빅데이터 3기

7 새로운 친구의 등장 – 리스트(난 다른 종류의 데이터와 함께할 수 있어)

7-1 다른 종류의 데이터를 리스트 형태로 만들어보기

7-2 리스트를 조금 자유롭게 다루어볼까요(1)

7-3 리스트를 알고 자유롭게 다루기(2)

7-4 음 잘하고 있어. 리스트는 더 친해졌군.

7-5 성별에 따른 키를 보고 싶다.(변수를 리스트로 만들어보기)

7-6 좀 더 쉽게 평균을 보자 (sapply)

7-7 남녀별 평균, 중앙값, 분산, 표준편차, 범위를 구해보자.

7-8 자 이제 도수분포표라는 것을 만들어볼까?

8 데이터가 조금 이상하다. 없는 값(결측치)이 있어요

8-1 결측치(없는 값)을 확인해 보자.

8-2 결측치(없는 값)을 제거해 보자. (no.omit(), complete.cases())

7 새로운 친구의 등장 – 리스트(난 다른 종류의 데이터와 함께할 수 있어)

7-1 리스트 데이터 넌 누구냐?

어떤 객체라도 받아들일 수 있는 데이터 객체

```
> DF <- read.csv("example_studentlist.csv")
> a <- c(1:20)
> s <- c("파스타", "짬뽕", "순두부찌개", "요거트", "커피")
> L <- c(T, F, F, T, T, T)
> is(DF)
[1] "data.frame" "list"       "oldClass"   "vector"
> is(a)
[1] "integer"           "numeric"
[3] "vector"            "data.frameRowLabels"
> is(s)
[1] "character"          "vector"
[3] "data.frameRowLabels" "SuperClassMethod"
> is(L)
[1] "logical" "vector"

> List
[[1]]
     name  sex age grade absence bloodtype height weight
1  김길동 남자  23     3      유         O  165.3   68.2
2  이미린 여자  22     2      무        AB  170.1   53.0
3  홍길동 남자  24     4      무         B  175.0   80.1
4  김철수 남자  23     3      무        AB  182.1   85.7
5  손세수 여자  20     1      유         A  168.0   49.5
6  박미희 여자  21     2      무         O  162.0   52.0
7  강수친 여자  22     1      무         O  155.2   45.3
8  이희수 여자  23     1      무         A  176.9   55.0
9  이철린 남자  23     3      무         B  178.5   64.2
10 방희철 남자  22     2      무         B  176.1   61.3
11 박수호 남자  24     4      유         O  167.1   62.0
12 임동민 남자  22     2      무        AB  180.0   75.8
13 김민수 남자  21     1      무         A  162.2   55.3
14 이희진 여자  23     3      무         O  176.1   53.1
15 김미진 여자  22     2      무         B  158.2   45.2
16 김동수 남자  24     4      유         B  168.6   70.2
17 여수근 남자  21     1      무         A  169.2   62.2

[[2]]
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19
[20] 20

[[3]]
[1] "파스타"     "짬뽕"       "순두부찌개" "요거트"
[5] "커피"
```

```
[[4]]
[1]  TRUE FALSE FALSE  TRUE  TRUE  TRUE

[[5]]
function (x, ...)
UseMethod("mean")
<bytecode: 0x00000000133b54a0>
<environment: namespace:base>
```

7-2 리스트를 조금 자유롭게 다루어볼까요(1)

리스트 항목 삭제

NULL을 이용하여 리스트의 항목을 삭제가능하다.

```
> List[1] <- NULL
> List
[[1]]
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19
[20] 20

[[2]]
[1] "파스타"    "짬뽕"      "순두부찌개" "요거트"
[5] "커피"

[[3]]
[1]  TRUE FALSE FALSE  TRUE  TRUE  TRUE

[[4]]
function (x, ...)
UseMethod("mean")
<bytecode: 0x00000000133b54a0>
<environment: namespace:base>
```

리스트 항목 선택

리스트를 선택할 수 있다.

```
> List[1]
[[1]]
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
[17] 17 18 19 20
```

리스트 생성

객체명 <- list(이름=객체, 이름=객체, 이름=객체).

```
# List를 이름넣어 만들기
ListNamed <- list(DataFrame1=DF, Num1=a, Char1=s, Logic1=L)
ListNamed
```

[실행결과]

```
> ListNamed <- list(DataFrame1=DF, Num1=a, Char1=s, Logic1=L)
> ListNamed
$DataFrame1
    name  sex age grade absence bloodtype height weight
1  김길동 남자  23     3      유         O  165.3   68.2
2  이미린 여자  22     2      무        AB  170.1   53.0
3  홍길동 남자  24     4      무         B  175.0   80.1
4  김철수 남자  23     3      무        AB  182.1   85.7
……
16 김동수 남자  24     4      유         B  168.6   70.2
17 여수근 남자  21     1      무         A  169.2   62.2

$Num1
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20

$Char1
[1] "파스타"     "짬뽕"       "순두부찌개" "요거트"     "커피"

$Logic1
[1]  TRUE FALSE FALSE  TRUE  TRUE  TRUE
```

리스트 생성

객체명 <- list(이름=객체, 이름=객체, 이름=객체).

```
# List를 이름넣어 만들기
ListNamed <- list(DataFrame1=DF, Num1=a, Char1=s, Logic1=L)
ListNamed
```

[실행결과]

```
> ListNamed <- list(DataFrame1=DF, Num1=a, Char1=s, Logic1=L)
> ListNamed
$DataFrame1
    name  sex age grade absence bloodtype height weight
1  김길동 남자  23     3      유         O  165.3   68.2
2  이미린 여자  22     2      무        AB  170.1   53.0
3  홍길동 남자  24     4      무         B  175.0   80.1
4  김철수 남자  23     3      무        AB  182.1   85.7
……
16 김동수 남자  24     4      유         B  168.6   70.2
17 여수근 남자  21     1      무         A  169.2   62.2

$Num1
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20

$Char1
[1] "파스타"     "짬뽕"       "순두부찌개" "요거트"     "커피"

$Logic1
[1]  TRUE FALSE FALSE  TRUE  TRUE  TRUE
```

리스트 항목 선택

사용 : 객체명["항목이름"]

```
# List 항목 선택
ListNamed["Num1"]
```

[실행결과]

```
> ListNamed["Num1"]
$Num1
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
```

리스트 항목 선택(2) – 두개를 선택할 수 없을까?

사용 : 객체명["항목이름"]

```
# List 항목 선택
ListNamed[c(열번호1, 열번호2)]
ListNamed[c(열번호1:열번호2)]
ListNamed[c("열이름1", "열이름2")]
```

[실행결과]

```
> ListNamed[c(3,4)]
$Char1
[1] "파스타"    "짬뽕"      "순두부찌개" "요거트"    "커피"

$Logic1
[1]  TRUE FALSE FALSE  TRUE  TRUE  TRUE

> ListNamed[c(2:4)]
$Num1
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20

$Char1
[1] "파스타"    "짬뽕"      "순두부찌개" "요거트"    "커피"

$Logic1
[1]  TRUE FALSE FALSE  TRUE  TRUE  TRUE

> ListNamed[c("Num1", "Logic1")]
$Num1
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20

$Logic1
[1]  TRUE FALSE FALSE  TRUE  TRUE  TRUE
```

리스트 항목 선택(2) – List[1], List[[1]], List$객체명의 차이?

List[1]은 리스트 객체의 첫번째 객체 리스트를 선택하는 것이다.

List[[1]]은 리스트 객체의 첫번째 객체 안의 첫번째 값(객체)을 선택하는 것이다.

```
# List 항목 선택
ListNamed[ [1] ]    # 리스트 (DataFrame이름) 선택
ListNamed[ 1 ]      # 데이터 프레임 선택

ListNamed[2]        # 리스트의 2번째 항목 선택
ListNamed[[2]]      # 리스트 2번째 항목의 값 선택
```

[실행결과]

```
> ListNamed[2]
$Num1
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20

> ListNamed[[2]]
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20

 > ListNamed$Num1
  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
 > ListNamed[[2]]
  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
```

7-3 리스트 항목 이름 바꾸기

하나의 항목 이름 바꾸기

사용법 : names(객체명)[항목번호] <- "이름"

```
# List 항목 선택
names(ListN)
names(ListN)[1] <- "DF1"
names(ListN)
```

현재 ListN의 항목 이름을 확인한다. 이후, 첫번째 항목에 대해 'DF1'으로 이름을 변경한다.

[실행결과]

```
> names(ListN)
[1] "DataFrame1" "Num1"       "Char1"      "Logic1"
> names(ListN)[1] <- "DF1"
> names(ListN)
[1] "DF1"    "Num1"   "Char1"  "Logic1"
```

여러 개의 항목 이름 바꾸기

사용법 : names(객체명)[항목번호] <- "이름"

```
# List 항목 선택
names(ListN)
names(ListN) <- c("DF2", "Num2", "Char2", "Logic2")
names(ListN)
```

항목의 이름의 개수만큼 이름을 넣어 해당 이름을 변경한다.

[실행결과]

```
> names(ListN)
[1] "DF1"    "Num1"   "Char1"  "Logic1"
> names(ListN) <- c("DF2", "Num2", "Char2", "Logic2")
> names(ListN)
[1] "DF2"    "Num2"   "Char2"  "Logic2
```

7-4 성별에 따른 키의 통계치를 보고 싶다.(변수를 리스트로 만들어보기)

많은 데이터를 가지고 있는데, 그중에 남녀만의 키의 통계를 한번 따로 구분해서 확인해 보고 싶은데 어떻게 해야 할까? 그렇다면 기준이 되는 것은 남녀가 될 것이다. 그리고 대상이 되는 값은 통계가 된다.

간단한 예제를 확인하고 접근해 보자.

정해진 기준에 따라 데이터를 다루어보기

사용법 : split(데이터, 명목형변수 기준)

```
# List 항목 선택
a <- c(80,90,70,90,95,60)
b <- c("Pass","Pass","Fail","Pass","Pass","Fail")
list1 <- split(a,b)
list1
```

Pass, Fail 의 기준으로 a의 값을 구분을 짓는다. 명목형 변수는 Pass, Fail이 된다.

[실행결과]

```
> a <- c(80,90,70,90,95,60)
> b <- c("Pass","Pass","Fail","Pass","Pass","Fail")
> split(a,b)
$Fail
[1] 70 60

$Pass
[1] 80 90 90 95

> list1 <- split(a,b)
> list1
$Fail
[1] 70 60

$Pass
[1] 80 90 90 95
> mean(list1[[1]])
[1] 65
> mean(list1[[2]])
[1] 88.75
```

7-6 좀 더 쉽게 평균을 보자 (sapply)

사용법 : sapply(데이터, 함수명)

```
# sapply를 이용한 평균구하기
sapply(list1, mean)

# sapply를 이용한 표준편차구하기
sapply(list1, sd)

# sapply를 이용한 범위 구하기
sapply(list1, range)
```

[실행결과]

```
> sapply(list1, mean)
 Fail  Pass
65.00 88.75
> sapply(list1, sd)
    Fail     Pass
7.071068 6.291529
> sapply(list1, range)
     Fail Pass
[1,]   60   80
[2,]   70   95
```

7-7 자 이제 도수분포표라는 것을 만들어볼까?

```
# 도수 분포표 만들어보기
# 01 데이터 불러오기
data <- read.csv("example_studentlist.csv")

# 02 혈액형을 나타내는 빈도수 구하기
Freq <- table(DF$bloodtype)
Freq

# 03 상대도수를 구하고 행으로 붙인다.
RFreq <- prop.table(Freq)
RFreq

# 04 행으로 붙이기
DF <- rbind(Freq, RFreq)
DF

# 05 행별 합을 구하기
DF <- addmargins(DF, margin=2)
DF
```

실행결과

```
> data <- read.csv("example_studentlist.csv")
> Freq <- table(DF$bloodtype)
> Freq

 A AB  B  O
 4  3  5  5

> ReleativeFreq <- prop.table(Freq)
> RFreq <- prop.table(Freq)
> RFreq

        A        AB         B         O
0.2352941 0.1764706 0.2941176 0.2941176

> DF <- rbind(Freq, RFreq)
> DF
             A        AB         B         O
Freq  4.0000000 3.0000000 5.0000000 5.0000000
RFreq 0.2352941 0.1764706 0.2941176 0.2941176
> DF
             A        AB         B         O Sum
Freq  4.0000000 3.0000000 5.0000000 5.0000000  17
RFreq 0.2352941 0.1764706 0.2941176 0.2941176   1
```

7-8 자 이제 도수분포표라는 것을 만들어볼까?(연속형변수)

```
# 도수 분포표 만들어보기(연속형 변수)
# 01 데이터 불러오기
data <- read.csv("example_studentlist.csv")

# 02 키의 값을 4구간으로 나누기
DivHeight <- cut(data$height, breaks=4)
DivHeight

# 03 구간별 개수(도수) 구하기
CntDivHeight <- table(DivHeight)
CntDivHeight

# 04 상대도수(상대적인값) 구하기
RFreq <- prop.table(CntDivHeight)
RFreq

# 05 구간별 도수와 상대도수 구하기
DataF <- rbind(CntDivHeight, RFreq)
DataF

# 06 행 이름 바꾸기
rownames(DataF)
rownames(DataF) <- c("도수", "상대도수")
DataF

# 07 누적상대도수도 구해보자.
CFreq <- cumsum(DataF[2,] )    # 2번째 행 선택 후, 누적상대도수 구하기
CFreq

# 08 누적상대도수 행 추가
DataF <- rbind(DataF, CFreq)
DataF

# 09 3번째 행 이름 변경
rownames(DataF)[3] <- "누적상대도수"
DataF

# 10 행의 합을 구해보기
DataF <- addmargins(DataF, margin=2)
DataF
```

실행결과

```
# 도수 분포표 만들어보기(연속형 변수)
# 01 데이터 불러오기
> data <- read.csv("example_studentlist.csv")

# 02 키의 값을 4구간으로 나누기
> DivHeight <- cut(data$height, breaks=4)
> DivHeight
 [1] (162,169] (169,175] (169,175] (175,182] (162,169] (162,169]
 [7] (155,162] (175,182] (175,182] (175,182] (162,169] (175,182]
[13] (162,169] (175,182] (155,162] (162,169] (169,175]
Levels: (155,162] (162,169] (169,175] (175,182]

#(설명) cut(나눌변수이름, break=나누고 싶은 구간의 개수, labels=나눈 구간의 이름)
#(EX) cut(weight, breaks=3, labels=c('1구간", "2구간", "3구간")

# 03 구간별 개수(도수) 구하기
> CntDivHeight <- table(DivHeight)
> CntDivHeight
DivHeight
(155,162] (162,169] (169,175] (175,182]
        2         6         3         6

# (설명) 각각의 구간에 대한 카운트(도수)를 구한다.
# 155~162 (155포함안함)이 2, 162~169 (162포함 안함) 6..으로 되어 있다.

# 04 상대도수(상대적인값) 구하기
> RFreq <- prop.table(CntDivHeight)
> RFreq
DivHeight
(155,162] (162,169] (169,175] (175,182]
0.1176471 0.3529412 0.1764706 0.3529412

# 05 구간별 도수와 상대도수 구하기
> DataF <- rbind(CntDivHeight, RFreq)
> DataF
            (155,162] (162,169] (169,175] (175,182]
CntDivHeight 2.0000000 6.0000000 3.0000000 6.0000000
RFreq        0.1176471 0.3529412 0.1764706 0.3529412

# 06 행 이름 바꾸기
> rownames(DataF)
[1] "CntDivHeight" "RFreq"
> rownames(DataF) <- c("도수", "상대도수")
> DataF
         (155,162] (162,169] (169,175] (175,182]
도수     2.0000000 6.0000000 3.0000000 6.0000000
상대도수 0.1176471 0.3529412 0.1764706 0.3529412
```

```
# 07 누적상대도수도 구해보자.
> CFreq <- cumsum(DataF[2,] )   # 2 번째 행 선택 후, 누적상대도수 구하기
> CFreq
(155,162] (162,169] (169,175] (175,182]
0.1176471 0.4705882 0.6470588 1.0000000


# 08 누적상대도수 행 추가
> DataF <- rbind(DataF, CFreq)
> DataF
        (155,162] (162,169] (169,175] (175,182]
도수     2.0000000 6.0000000 3.0000000 6.0000000
상대도수 0.1176471 0.3529412 0.1764706 0.3529412
CFreq    0.1176471 0.4705882 0.6470588 1.0000000


# 09 3번째 행 이름 변경
> rownames(DataF)[3] <- "누적상대도수"
> DataF
            (155,162] (162,169] (169,175] (175,182]
도수         2.0000000 6.0000000 3.0000000 6.0000000
상대도수     0.1176471 0.3529412 0.1764706 0.3529412
누적상대도수 0.1176471 0.4705882 0.6470588 1.0000000


# 10 행의 합을 구해보기
> DataF <- addmargins(DataF, margin=2)
> DataF
            (155,162] (162,169] (169,175] (175,182]      Sum
도수         2.0000000 6.0000000 3.0000000 6.0000000 17.000000
상대도수     0.1176471 0.3529412 0.1764706 0.3529412  1.000000
누적상대도수 0.1176471 0.4705882 0.6470588 1.0000000  2.235294
```