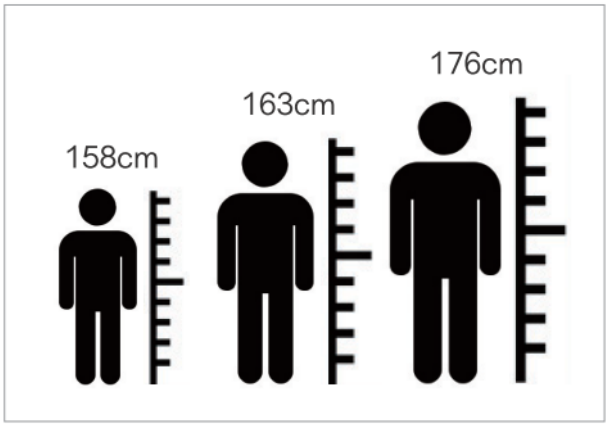
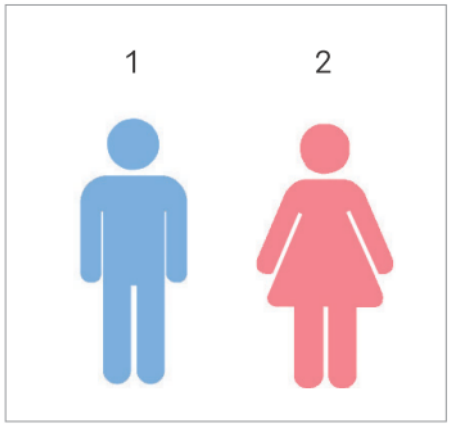
02. R 내장 함수, 변수 타입과 데이터 구조

연속 변수



범주 변수



02-1. R 내장 함수로 데이터 추출하기

행 번호로 행 추출하기

데이터 준비하기

```
exam <- read.csv("csv_exam.csv")
```

행 번호로 행 추출하기

대괄호안 쉼표 기준, 왼쪽에 행 번호(인덱스) 입력

- 인덱스(Index) : 데이터의 위치 또는 순서를 의미하는 값
- 인덱싱(Indexing) : 인덱스를 이용해 데이터를 추출하는 작업

```
exam[]    # 조건 없이 전체 데이터 출력
```

```
##    id class math english science
## 1   1     1   50      98      50
## 2   2     1   60      97      60
## 3   3     1   45      86      78
## 4   4     1   30      98      58
## 5   5     2   25      80      65
## 6   6     2   50      89      98
## 7   7     2   80      90      45
## 8   8     2   90      78      25
## 9   9     3   20      98      15
## 10 10     3   50      98      45
## 11 11     3   65      65      65
## 12 12     3   45      85      32
## 13 13     4   46      98      65
## 14 14     4   48      87      12
## 15 15     4   75      56      78
## 16 16     4   58      98      65
```

```
## 17 17     5   65       68       98
## 18 18     5   80       78       90
## 19 19     5   89       68       87
## 20 20     5   78       83       58
```

```r
exam[1,]  # 1행 추출

##   id class math english science
## 1  1     1   50      98      50

exam[2,]  # 2행 추출

##   id class math english science
## 2  2     1   60      97      60
```

조건을 충족하는 행 추출하기

```
exam[exam$class == 1,]  # class가 1인 행 추출
```

```
##   id class math english science
## 1  1     1   50      98      50
## 2  2     1   60      97      60
## 3  3     1   45      86      78
## 4  4     1   30      98      58
```

```
exam[exam$math >= 80,]  # 수학점수가 80점 이상인 행 추출
```

```
##    id class math english science
## 7   7     2   80      90      45
## 8   8     2   90      78      25
## 18 18     5   80      78      90
## 19 19     5   89      68      87
```

```r
# 1반 이면서 수학점수가 50점 이상
exam[exam$class == 1 & exam$math >= 50,]
```

```
##   id class math english science
## 1  1     1   50      98      50
## 2  2     1   60      97      60
```

```r
# 영어점수가 90점 미만이거나 과학점수가 50점 미만
exam[exam$english < 90 | exam$science < 50,]
```

```
##    id class math english science
## 3   3     1   45      86      78
## 5   5     2   25      80      65
## 6   6     2   50      89      98
## 7   7     2   80      90      45
## 8   8     2   90      78      25
## 9   9     3   20      98      15
## 10 10     3   50      98      45
## 11 11     3   65      65      65
## 12 12     3   45      85      32
## 14 14     4   48      87      12
## 15 15     4   75      56      78
## 17 17     5   65      68      98
## 18 18     5   80      78      90
## 19 19     5   89      68      87
## 20 20     5   78      83      58
```

열 번호로 변수 추출하기

대괄호안 쉼표 오른쪽에 조건을 입력

```
exam[,1]  # 첫 번째 열 추출
```

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

```
exam[,2]  # 두 번째 열 추출
```

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

```
exam[,3]  # 세 번째 열 추출
```

```
##  [1] 50 60 45 30 25 50 80 90 20 50 65 45 46 48 75 58 65 80 89 78
```

변수명으로 변수 추출하기

```
exam[, "class"]  # class 변수 추출

##  [1] 1 1 1 1 2 2 2 2 3 3 3 3 4 4 4 4 5 5 5 5

exam[, "math"]   # math 변수 추출

##  [1] 50 60 45 30 25 50 80 90 20 50 65 45 46 48 75 58 65 80 89 78

exam[,c("class", "math", "english")]  # class, math, english 변수 추출

##    class math english
## 1      1   50      98
## 2      1   60      97
## 3      1   45      86
## 4      1   30      98
## 5      2   25      80
## 6      2   50      89
## 7      2   80      90
## 8      2   90      78
## 9      3   20      98
## 10     3   50      98
## 11     3   65      65
## 12     3   45      85
## 13     4   46      98
## 14     4   48      87
## 15     4   75      56
```

```
## 16      4    58       98
## 17      5    65       68
## 18      5    80       78
## 19      5    89       68
## 20      5    78       83
```

행, 변수 동시 추출하기

```
# 행, 변수 모두 인덱스
exam[1,3]
```

```
## [1] 50
```

```
# 행 인덱스, 열 변수명
exam[5, "english"]
```

```
## [1] 80
```

```
# 행 부등호 조건, 열 변수명
exam[exam$math >= 50, "english"]
```

```
##  [1] 98 97 89 90 78 98 65 56 98 68 78 68 83
```

```
# 행 부등호 조건, 열 변수명
exam[exam$math >= 50, c("english", "science")]
```

```
##    english science
## 1       98      50
## 2       97      60
## 6       89      98
## 7       90      45
## 8       78      25
## 10      98      45
## 11      65      65
```

```
## 15       56       78
## 16       98       65
## 17       68       98
## 18       78       90
## 19       68       87
## 20       83       58
```

dplyr과 내장 함수의 차이

문제) 수학 점수 50 이상, 영어 점수 80 이상인 학생들을 대상으로 각 반의 전 과목 총평균을 구하라.

내장 함수 코드

```
exam$tot <- (exam$math + exam$english + exam$science)/3
aggregate(data=exam[exam$math >= 50 & exam$english >= 80,], tot~class, mean)
```

dplyr 코드

```
exam %>%
  filter(math >= 50 & english >= 80) %>%
  mutate(tot = (math + english + science)/3) %>%
  group_by(class) %>%
  summarise(mean = mean(tot))
```

혼자서 해보기

`mpg` 데이터를 이용해서 분석 문제를 해결해 보세요.

아래는 `dplyr` 패키지 함수들을 이용해 `"compact"`와 `"suv"` 차종의 '도시 및 고속도로 통합 연비' 평균을 구하는 코드입니다.

```
mpg <- as.data.frame(ggplot2::mpg)                    # mpg 데이터 불러오기

mpg %>%
  mutate(tot = (cty + hwy)/2) %>%                     # 통합 연비 변수 생성
  filter(class == "compact" | class == "suv") %>%     # compact, suv 추출
  group_by(class) %>%                                 # class 별 분리
  summarise(mean_tot = mean(tot))                     # tot 평균 산출

## # A tibble: 2 x 2
##     class mean_tot
##     <chr>    <dbl>
## 1 compact 24.21277
## 2     suv 15.81452
```

Q1. `dplyr` 대신 R 내장 함수를 이용해 `"suv"`와 `"compact"`의 '도시 및 고속도로 통합 연비' 평균을 구해보세요.

힌트

우선 `cty`와 `hwy`를 이용해 '통합 연비 변수'를 만드세요. 그런 다음, `class`가 `"compact"`인 행과 `"suv"`인 행을 추출해 두 종류의 데이터를 만드세요. 이렇게 만든 두 데이터를 이용해 통합 연비 변수 평균을 각각 구하면 됩니다.

정답

Q1. `dplyr` 대신 R 내장 함수를 이용해 `"suv"`와 `"compact"`의 '도시 및 고속도로 통합 연비' 평균을 구해보세요.

```
mpg$tot <- (mpg$cty + mpg$hwy)/2          # 통합 연비 변수 만들기

df_comp <- mpg[mpg$class == "compact",]   # compact 추출
df_suv <- mpg[mpg$class == "suv",]        # suv 추출

mean(df_comp$tot)                         # compact 의 tot 평균 산출
```

```
## [1] 24.21277
```

```
mean(df_suv$tot)                          # suv 의 tot 평균 산출
```

```
## [1] 15.81452
```

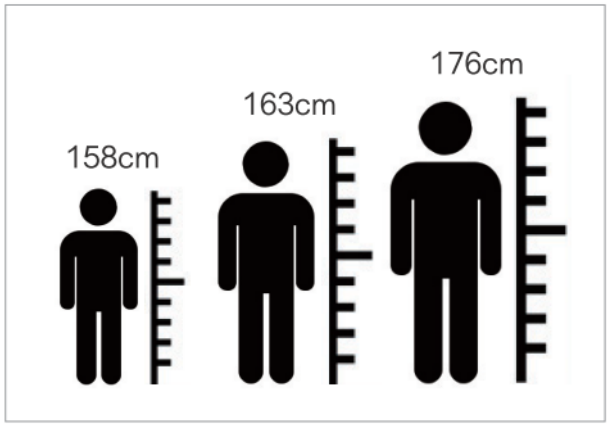
02-2. 변수 타입

변수에는 여러 가지 타입(Type, 속성)이 있음

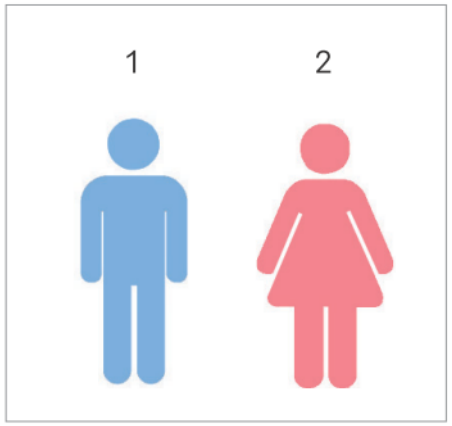
- 함수에 따라 적용 가능한 변수 타입 다름
- 분석 전에 변수 타입이 무엇인지 확인 필요
- 함수 실행했을 때 오류 발생 또는 예상과 다른 결과가 출력되면 변수 타입 확인 후 함수에 맞게 변경

변수의 종류

연속 변수



범주 변수



- **1. 연속 변수(Continuous Variable) - Numeric 타입**
  - 값이 연속적이고 크기를 의미
  - 더하기 빼기, 평균 구하기 등 산술 가능
  - ex) 키, 몸무게, 소득
- **2. 범주 변수(Categorical Variable) - Factor 타입**
  - 값이 대상을 분류하는 의미를 지님
  - 산술 불가능
  - ex) 성별, 거주지

변수	Data Type	예
연속 변수	Numeric	키(..., 151, 152, ...), 몸무게(..., 58, 59, ...)
범주 변수	Factor	성별(1, 2), 지역(1, 2, 3, 4)

변수 타입 간 차이 알아보기

```
var1 <- c(1,2,3,1,2)           # numeric 변수 생성
var2 <- factor(c(1,2,3,1,2))   # factor 변수 생성

var1  # numeric 변수 출력

## [1] 1 2 3 1 2

var2  # factor 변수 출력

## [1] 1 2 3 1 2
## Levels: 1 2 3
```

```
var1+2  # numeric 변수로 연산
```

```
## [1] 3 4 5 3 4
```

```
var2+2  # factor 변수로 연산
```

```
## Warning in Ops.factor(var2, 2): '+' not meaningful for factors
```

```
## [1] NA NA NA NA NA
```

변수 타입 확인하기

```
class(var1)

## [1] "numeric"

class(var2)

## [1] "factor"
```

factor 변수의 구성 범주 확인하기

```
levels(var1)

## NULL

levels(var2)

## [1] "1" "2" "3"
```

문자로 구성된 factor 변수

```r
var3 <- c("a", "b", "b", "c")           # 문자 변수 생성
var4 <- factor(c("a", "b", "b", "c"))  # 문자로 된 factor 변수 생성

var3
```

```
## [1] "a" "b" "b" "c"
```

```r
var4
```

```
## [1] a b b c
## Levels: a b c
```

```r
class(var3)
```

```
## [1] "character"
```

```r
class(var4)
```

```
## [1] "factor"
```

함수마다 적용 가능한 변수 타입이 다르다

```
mean(var1)

## [1] 1.8

mean(var2)

## Warning in mean.default(var2): argument is not numeric or logical:
## returning NA

## [1] NA
```

변수 타입 바꾸기

```r
var2 <- as.numeric(var2)  # numeric 타입으로 변환
mean(var2)                # 함수 재적용

## [1] 1.8

class(var2)               # 타입 확인

## [1] "numeric"

levels(var2)              # 범주 확인

## NULL
```

변환 함수(Coercion Function)

함수	기능
as.numeric()	numeric으로 변환
as.factor()	factor로 변환
as.character()	character로 변환
as.Date()	Date로 변환
as.data.frame()	Data Frame으로 변환

혼자서 해보기

`mpg` 데이터의 `drv` 변수는 자동차의 구동 방식을 나타냅니다. `mpg` 데이터를 이용해 아래 문제를 해결해 보세요.

- Q1. `drv` 변수의 타입을 확인해 보세요.
- Q2. `drv` 변수를 `as.factor()`를 이용해 `factor` 타입으로 변환한 후 다시 타입을 확인해 보세요.
- Q3. `drv`가 어떤 범주로 구성되는지 확인해 보세요.

정답

```r
class(mpg$drv)                    # 타입 확인
```

```
## [1] "character"
```

```r
mpg$drv <- as.factor(mpg$drv)     # factor로 변환
class(mpg$drv)                    # 타입 확인
```

```
## [1] "factor"
```

```r
levels(mpg$drv)                   # 범주 확인
```

```
## [1] "4" "f" "r"
```

02-3. 데이터 구조

- 데이터 프레임 외에도 다양한 데이터 구조가 있음
- 데이터 구조에 따라 활용 방법 다름

데이터 구조	차원	특징
벡터(Vactor)	1차원	한 가지 변수 타입으로 구성
데이터 프레임(Data Frame)	2차원	다양한 변수 타입으로 구성
매트릭스(Matrix)	2차원	한 가지 변수 타입으로 구성
어레이(Array)	다차원	2차원 이상의 매트릭스
리스트(List)	다차원	서로 다른 데이터 구조 포함

데이터 구조 비교하기

1. 벡터(Vactor)

- 하나 또는 여러 개의 값으로 구성된 데이터 구조
- 여러 타입을 섞을 수 없고, 한 가지 타입으로만 구성 가느

```
# 벡터 만들기
a <- 1
a

## [1] 1

b <- "hello"
b

## [1] "hello"

# 데이터 구조 확인
class(a)

## [1] "numeric"

class(b)

## [1] "character"
```

2. 데이터 프레임(Data Frame)

- 행과 열로 구성된 2차원 데이터 구조
- 다양한 변수 타입으로 구성 가능

```
# 데이터 프레임 만들기
x1 <- data.frame(var1 = c(1,2,3),
                 var2 = c("a","b","c"))
x1

##   var1 var2
## 1    1    a
## 2    2    b
## 3    3    c

# 데이터 구조 확인
class(x1)

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

3. 매트릭스(Matrix)

- 행과 열로 구성된 2차원 데이터 구조
- 한 가지 타입으로만 구성 가능

```
# 매트릭스 만들기 - 1~12 로 2 열
x2 <- matrix(c(1:12), ncol = 2)
x2
```

```
##      [,1] [,2]
## [1,]    1    7
## [2,]    2    8
## [3,]    3    9
## [4,]    4   10
## [5,]    5   11
## [6,]    6   12
```

```
# 데이터 구조 확인
class(x2)
```

```
## [1] "matrix"
```

4. 어레이(Array)

- 2차원 이상으로 구성된 매트릭스
- 한 가지 타입으로만 구성 가능

```
# array 만들기 - 1~20 으로 2 행 x 5 열 x 2 차원
x3 <- array(1:20, dim = c(2, 5, 2))
x3
```

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

5. 리스트(List)

- 모든 데이터 구조를 포함하는 데이터 구조
- 여러 데이터 구조를 합해 하나의 리스트로 구성 가능

```
# 리스트 생성 - 앞에서 생성한 데이터 구조 활용
x4 <- list(f1 = a,    # 벡터
           f2 = x1,   # 데이터 프레임
           f3 = x2,   # 매트릭스
           f4 = x3)   # 어레이
x4

## $f1
## [1] 1
##
## $f2
##   var1 var2
## 1    1    a
## 2    2    b
## 3    3    c
##
## $f3
##      [,1] [,2]
## [1,]    1    7
## [2,]    2    8
```

```
## [3,]    3    9
## [4,]    4   10
## [5,]    5   11
## [6,]    6   12
##
## $f4
## , , 1
##
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    3    5    7    9
## [2,]    2    4    6    8   10
##
## , , 2
##
##      [,1] [,2] [,3] [,4] [,5]
## [1,]   11   13   15   17   19
## [2,]   12   14   16   18   20
```

```
# 데이터 구조 확인
class(x4)
```

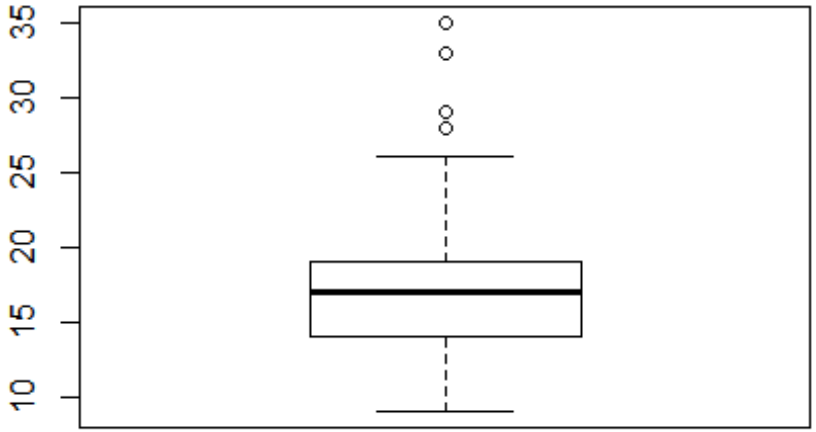
```
## [1] "list"
```

리스트 활용

- 함수의 결과물이 리스트 형태로 반환되는 경우 많음
- 리스트를 활용하면 함수의 결과물에서 특정 값을 추출 가능

boxplot() 출력 결과물에서 값 추출하기

```
mpg <- ggplot2::mpg
x <- boxplot(mpg$cty)
```



```
x

## $stats
##      [,1]
## [1,]    9
## [2,]   14
## [3,]   17
## [4,]   19
## [5,]   26
## attr(,"class")
##         1
## "integer"
##
## $n
## [1] 234
##
## $conf
##          [,1]
## [1,] 16.48356
## [2,] 17.51644
##
## $out
## [1] 28 28 33 35 29
##
## $group
## [1] 1 1 1 1 1
##
```

```
## $names
## [1] "1"
```

```r
x$stats[,1]     # 요약 통계량 추출

## [1]  9 14 17 19 26

x$stats[,1][3]  # 중앙값 추출

## [1] 17

x$stats[,1][2]  # 1분위수 추출

## [1] 14
```

정리하기

```r
# 1.데이터 추출하기
exam[1,]                                    # 행 번호로 행 추출
exam[exam$class == 1,]                      # 조건을 충족하는 행 추출
exam[exam$class == 1 & exam$math >= 50,]    # 여러 조건을 충족하는 행 추출

exam[,1]                                    # 열 번호로 변수 추출
exam[, "class"]                             # 변수명으로 변수 추출
exam[,c("class", "math", "english")]        # 변수명으로 여러 변수 추출
exam[1,3]                                   # 행, 변수 동시 추출 - 인덱스
exam[exam$math >= 50, "english"]            # 행, 변수 동시 추출 - 조건문, 변수명


# 2.변수 타입
var <- c(1,2,3,1,2)                     # numeric 변수 만들기
var <- factor(c(1,2,3,1,2))             # factor 변수 만들기
var <- factor(c("a", "b", "b", "c"))    # 문자로 구성된 factor 변수 만들기

class(var)                              # 변수 타입 확인하기
levels(var)                             # factor 변수의 구성 범주 확인
var <- as.numeric(var)                  # factor 타입을 numeric 타입으로 변환하기
```

정리하기

```r
# 3.데이터 구조
a <- 1                                          # 벡터 만들기
b <- "hello"

x1 <- data.frame(var1 = c(1,2,3),               # 데이터 프레임 만들기
                 var2 = c("a","b","c"))

x2 <- matrix(c(1:12), ncol = 2)                 # 매트릭스 만들기

x3 <- array(1:20, dim=c(2, 5, 2))               # 어레이 만들기

x4 <- list(f1 = a,                              # 리스트 만들기
           f2 = x1,
           f3 = x2,
           f4 = x3)

# 리스트 활용하기
x <- boxplot(mpg$cty)  # 상자 그림 만들기
x$stats[,1]            # 요약 통계량 추출
```