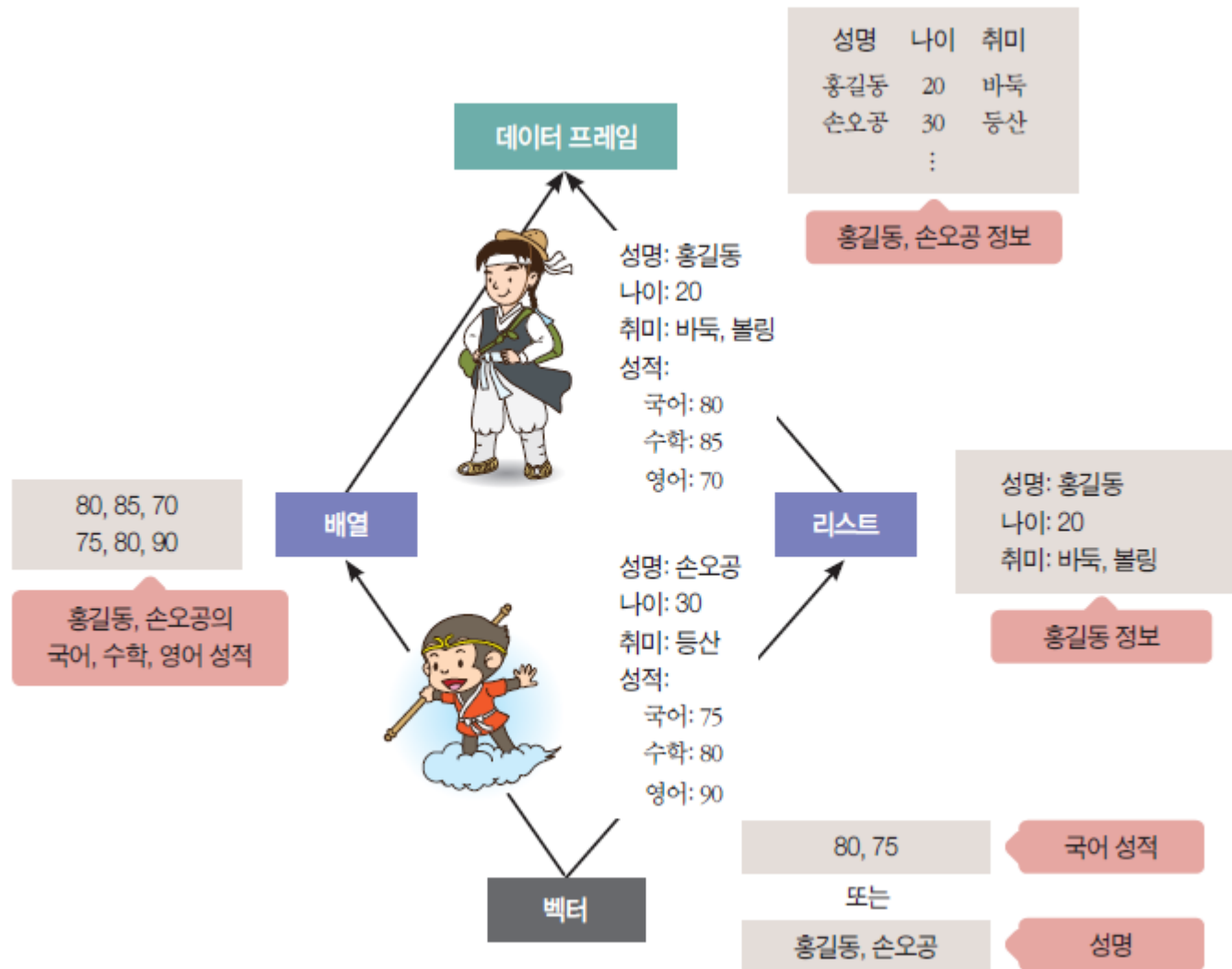




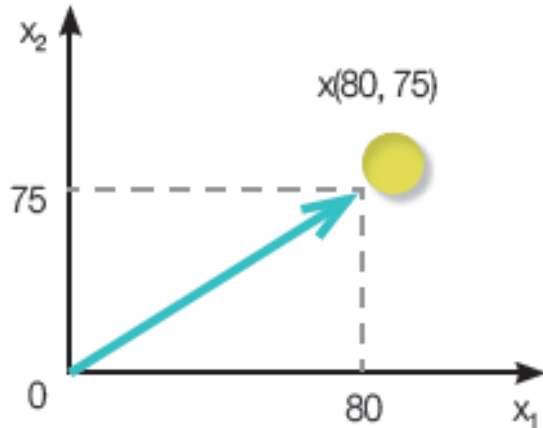
# 실습 2강

## 데이터 구조와 R을 이용한 데이터 분석 기초

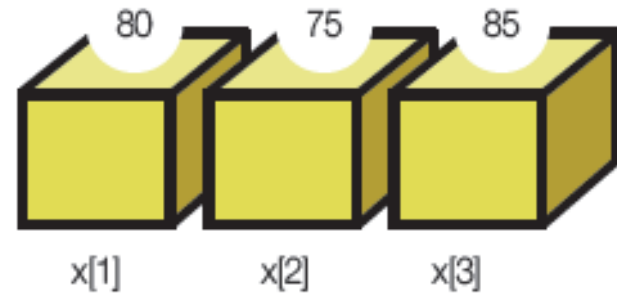
# 데이터 구조의 이해



# 벡터와 연산자



두 원소로 구성되는 x 벡터



x 벡터의 각 원소 값

## 벡터 만들기

```
x <- c(80, 85, 70)
```

```
x
```

홍길동의 국어, 수학, 영어 성적을 벡터로 저장

객체명을 입력하면 객체에  
할당된 데이터가 출력됨

출력 결과

```
[1] 80 85 70
```



# 산술 연산자

## ❖ 사칙연산과 응용

유형	기호	예	결과
더하기	+	2+3	5
빼기	-	10-3	7
곱하기	*	3*4	12
나누기	/	8/2	4
거듭제곱	^ 또는 **	2^3	8
나머지	%%	10%%3	1
몫	%/%	10%/3	3



# 비교 연산자

## ❖ 값들의 크기를 비교

유형	기호	예	결과
작음	<	$x < -3$ $x < 10$	TRUE
이하	<=	$x < -3$ $x <= 10$	TRUE
큼	>	$x < -3$ $x > 10$	FALSE
이상	>=	$x < -3$ $x >= 10$	FALSE
같음	==	$x < -3$ $x == 10$	FALSE
같지 않음	!=	$x < -3$ $x != 10$	TRUE



# 벡터의 원소

❖ 벡터를 구성하는 특정 원소 값들을 출력하거나, 수정하는 과정

`x <- c(1, 2, 3, 4, 5)`

`x[2]`

# 두 번째 원소

`x[c(1, 3, 5)]`

# 1, 3, 5번째 원소

`x[-c(2, 4)]`

# 2, 4번째 제외한 원소

`x[x > 2]`

# 원소의 값이 2보다 큰 값들만 출력

`x[x >= 2 & x <= 4]`

# 원소의 값이 2 이상이고 4 이하인 값들만 출력

`x[2] <- 20`

# 2번째 원소의 값을 20으로 수정

`x[c(3, 4)] <- 15`

# 3, 4번째 원소들의 값을 모두 15로 수정

`x[x <= 15] <- 10`

# 15 이하인 원소 값들을 모두 10으로 수정



# 함수 이용 벡터 연산

## ❖ 벡터 값에 대한 함수의 활용 예

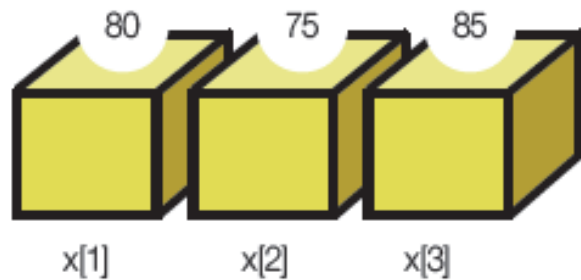
```
x <- seq(1:10)
```

mean(x)	# x 벡터의 원소 값들의 평균
var(x)	# x 벡터의 원소 값들의 분산
sd(x)	# x 벡터의 원소 값들의 표준편차
sqrt(x)	# x 벡터의 원소 값들의 제곱근
length(x)	# x 벡터의 원소 값들의 개수
abs(x)	# x 벡터의 원소 값들의 절대값

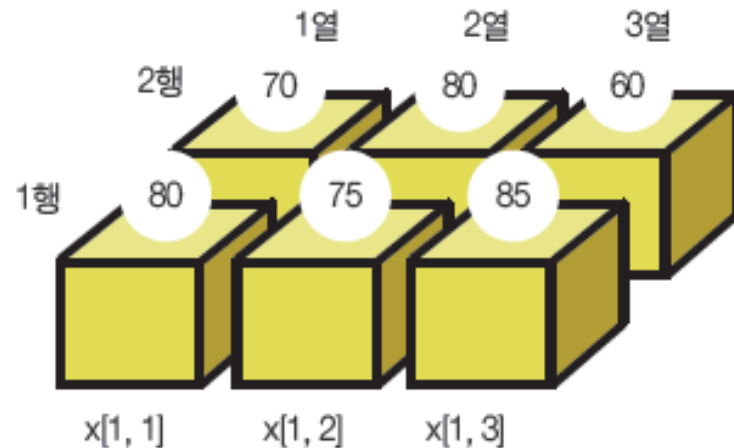


# 배열과 행렬

- ❖ 배열: 한 개 이상의 벡터로 구성, 동일한 데이터 유형
  - 1차원 배열: 벡터
  - 2차원 배열: 행과 열, 행렬
  - 함수: `array()`



1차원 배열



2차원 배열





# 1차원 배열

```
x <- array(1:3, dim=c(3))
```

```
x
```

1~3까지의 정수를 1행 3열의 1차원으로 표시  
☞ x <- seq(1:3)과 동일

출력 결과

```
[1] 1 2 3
```

array()

패키지	base
사용법	<b>array</b> (data, dim, [dimnames])
설명	배열 생성
반환 값	배열

매개변수	설명
data	벡터 자료
dim	차원을 나타내는 벡터
dimnames	각 차원의 이름을 나타내는 벡터



# 2차원 배열

```
x <- array(1:6, dim=c(2, 3))
```

```
x
```

1~6까지의 정수를 2행 3열의 2차원으로 표시  
⇒ 열 단위의 순서로 데이터를 할당

출력결과

```
      [,1] [,2] [,3]  
[1,]  1    3    5  
[2,]  2    4    6
```

```
x[1, 3]
```

```
x[, 3]
```

```
x[, -3]
```

```
x[1, 2] <- 20
```

```
names <-
```

```
  list(c("1행", "2행"),  
       c("1열", "2열", "3열"))
```

# 1행 3열 값

# 3열의 모든 값들

# 3열을 제외한 모든 열의 값들

# 1행 2열의 값을 20으로 수정

# 행과 열 이름을 갖는 두 벡터로  
하는 리스트 생성



# 행렬

행렬: 행과 열로 구성되는 2차원 배열

함수: `matrix()`

```
x <- matrix(1:6, nrow=2)
```

x

1~6 사이의 정수를 행의 수가 2인 행렬로 만들  
☞ 원소 값의 할당 순서는 열 기준임

출력 결과

```
      [,1] [,2] [,3]  
[1,] 1    3    5  
[2,] 2    4    6
```

`matrix()`

패키지	base
사용법	<code>matrix(data, nrow, ncol byrow, [dimnames])</code>
설명	행렬 생성
반환 값	행렬

매개변수	설명
data	벡터 자료
nrow	행 요소의 개수
ncol	열 요소의 개수
byrow	data를 행 단위로 배치할지에 대한 여부. 디폴트는 FALSE이며 열 단위로 배치
dimnames	행과 열의 이름 list



# 리스트

- ❖ 벡터의 각 원소: 이름, 서로 다른 데이터 유형 가능, 하나 이상의 값 가능
- ❖ 함수: list()



```
x <- list("홍길동", "2016001", 20, c("IT융합", "데이터 관리"))  
x
```

‘홍길동’을 표현하는 리스트  
⇨ 객체들의 값만 있는 경우

## 출력 결과

```
[[1]]  
[1] "홍길동" (첫 번째 객체와 원소 값)  
  
[[2]]  
[1] "2016001" (두 번째 객체와 원소 값)  
  
[[3]]  
[1] 20 (세 번째 객체와 원소 값)  
  
[[4]]  
[1] "IT융합" "데이터 관리" (네 번째 객체와 원소 값들)
```

```
y <- list("성명"="홍길동", "학번"="2016001", "나이"=20, "수강과목"=c("IT융합",  
"데이터 관리"))  
y
```

각 원소 값에 이름이 주어짐  
⇨ ‘성명’ 원소는 ‘홍길동’의 값을 가짐

## 출력 결과

```
$성명  
[1] "홍길동" (첫 번째 객체(성명)와 원소 값)
```



# 데이터 프레임

- ❖ 동일한 속성들을 가지는 여러 개체들로 구성
- ❖ 2차원적인 데이터 구조
- ❖ 속성들 간 데이터 유형은 서로 다를 수 있음



# 데이터 프레임 만들기

```
x <- data.frame(성명=c("홍길동", "손오공"), 나이=c(20, 30), 주소=c("서울", "부산"))  
x
```

두 객체에 해당  
하는 데이터 프  
레이م

## 출력 결과

	성명	나이	주소	
1	홍길동	20	서울	홍길동 객체
2	손오공	30	부산	손오공 객체

## data.frame()

패키지	base
사용법	<b>data.frame</b> (..., row.names=NULL)
설명	데이터 프레임 생성
반환 값	데이터 프레임

매개변수	설명
...	값 또는 태그=값
row.names	열의 이름



# 행/열 추가와 요소값 보기

```
x <- cbind(x, 학과=c("전산학", "경영학"))
```

데이터 프레임 x에 열(학과) 추가

출력 결과

	성명	나이	주소	학과
1	홍길동	20	서울	전산학
2	손오공	30	부산	경영학

```
x <- rbind(x, data.frame(성명="장발장", 나이=40, 주소="파리", 학과="전산학"))
```

데이터 프레임 x에 행(장발장) 추가

출력 결과

	성명	나이	주소	학과
1	홍길동	20	서울	전산학
2	손오공	30	부산	경영학
3	장발장	25	파리	전산학

# 3행 2열의 요소 값  
x[3, 2]

# 3행의 모든 값  
x[3, ]

# 2행을 제외한 모든 값  
x[-2,]

# "성명" 요소  
# x[1]과 동일한 결과  
x["성명"]

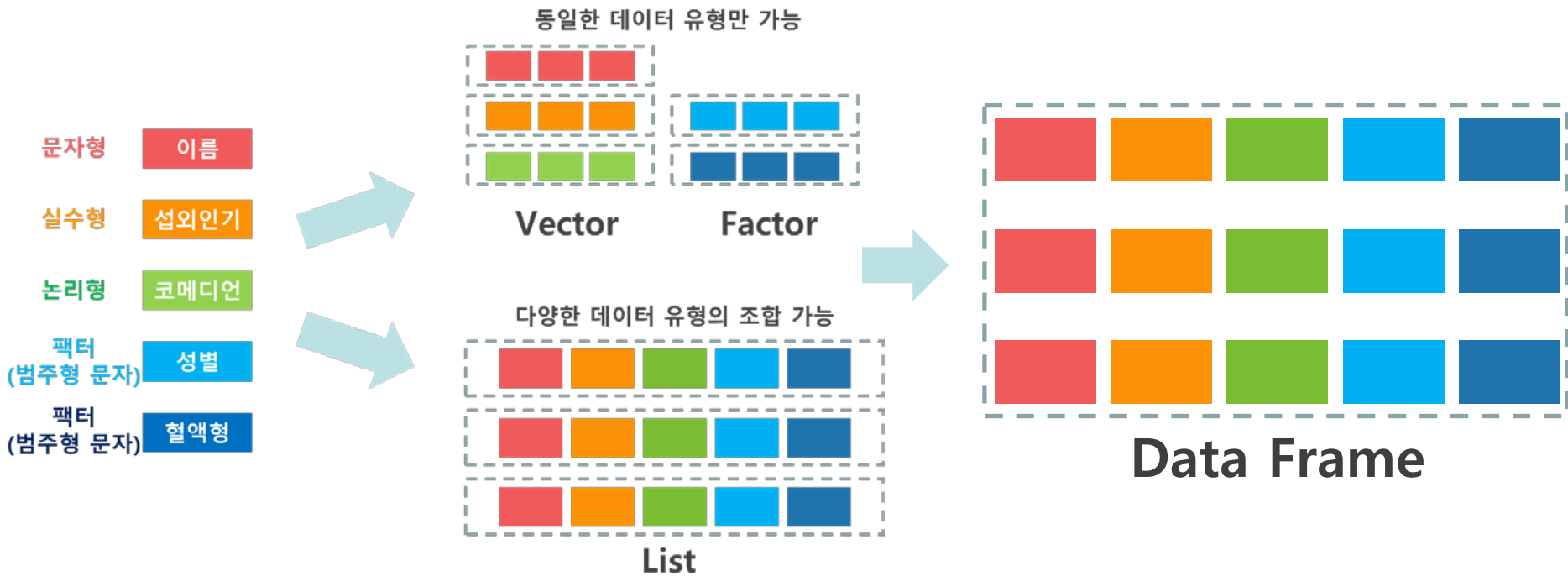
# "성명" 요소값  
x\$성명, x[["성명"]], x[[1]]

# 1열 요소 값에서 두 번째 값  
x[[1]][2]



# R의 데이터 구조

## ❖ 데이터 구조

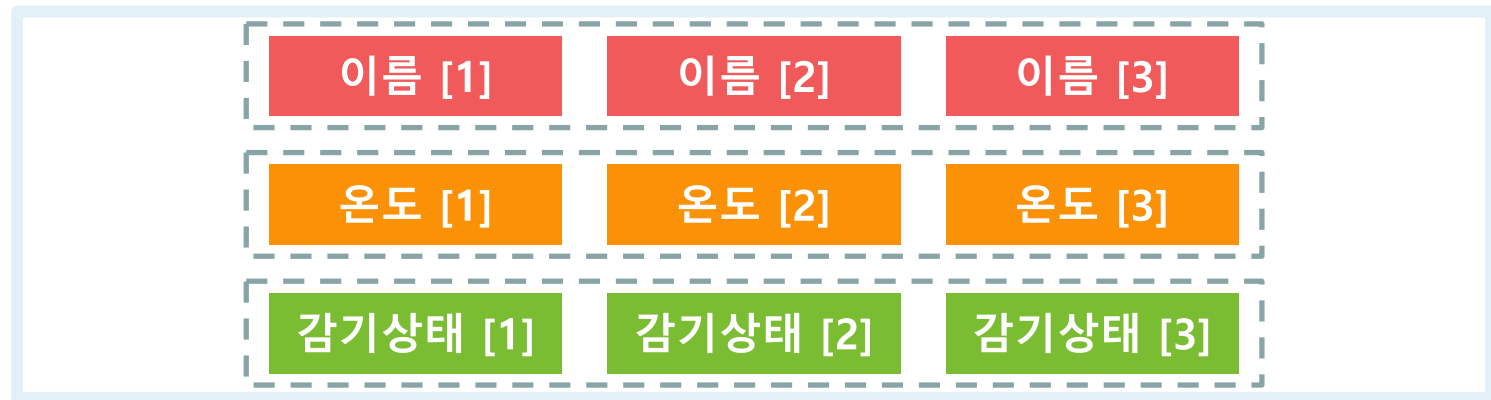




# R의 데이터 구조

## ❖ 벡터 (Vector)

- R에서 가장 작은 데이터 단위
- 4가지 유형의 벡터가 있음



● 데이터 구조

```
subject_name <- c("John", "Dunpy", "Steve")  
temperature <- c(32.2, 33.0, 37.9)  
flu_status <- c(FALSE, FALSE, TRUE)
```

● R 코드

# R의 데이터 구조

## 실수형 (Numeric)

```
> temperature <- c(32.2, 33.0, 37.9)
> is(temperature)
[1] "numeric" "vector"
```

## 논리형 (Logical)

```
> flu_status <- c(FALSE, FALSE, TRUE)
> is(flu_status)
[1] "logical" "vector"
```

## 문자형 (Character)

```
> subject_name <- c("John", "Dunpy", "Steve")
> is(subject_name)
[1] "character"
[2] "vector"
[3] "data.frameRowLabels"
[4] "superClassMethod"
```

# R의 데이터 구조

## ❖ 팩터 (factor)

- 명목형 변수를 대표하기 위해 사용되는 **특별한 용도의 벡터** (순서형도 가능)
- 범주형 라벨을 한번만 저장하여 사용
- "Levels=" 구문을 통하여 명시 가능

```
> gender <- factor(c("MALE", "FEMALE", "MALE"))  
> gender  
[1] MALE    FEMALE MALE  
Levels: FEMALE MALE
```

- R 코드 (명목형)

```
> gender <- factor(gender, ordered=T)  
> gender  
[1] MALE    FEMALE MALE  
Levels: FEMALE < MALE
```

- R 코드 (순서형)

```
> blood <- factor(c("O", "AB", "A"),  
+               levels = c("A", "B", "AB", "O"))  
> blood  
[1] O  AB A  
Levels: A B AB O
```

- R 코드 (명목형 직접 만들기)

# R의 데이터 구조

## ❖ 데이터 구조

문자형

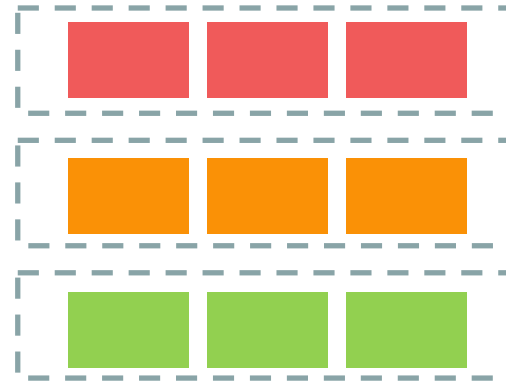
이름

실수형

온도

논리형

감기상태



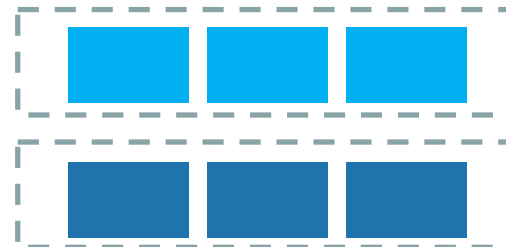
Vector

팩터  
(범주형/문자)

성별

팩터  
(범주형/문자)

혈액형



Factor

범주형 벡터

# R의 데이터 구조

## ❖ 리스트 (list)

- 벡터의 특별한 다른 형태로써, **자유로운 객체 사용** 가능
- 모든 원소가 같은 타입이어야 하는 벡터와 달리 타입이 달라도 되는 유연성 있음
- 기계 학습 모델의 설정 매개변수의 집합 사용



List 데이터 구조

# R의 데이터 구조

## ❖ 리스트 (list)

```
John <- list(fullname = subject_name[1],  
             temperature = temperature[1],  
             flu_status = flu_status[1],  
             gender = gender[1],  
             blood = blood[1])
```

● R 코드 (아이템 순서에 따른 list 만들기)

# R의 데이터 구조

## ❖ 리스트 (list)

<pre>&gt; John \$fullname [1] "John"</pre>	----->	<pre>&gt; John[1] \$fullname [1] "John"</pre>
<pre>\$temperature [1] 32.2</pre>	----->	<pre>&gt; John[2] \$temperature [1] 32.2</pre>
<pre>\$flu_status [1] FALSE</pre>	----->	<pre>&gt; John[3] \$flu_status [1] FALSE</pre>
<pre>\$gender [1] MALE Levels: FEMALE &lt; MALE</pre>	----->	<pre>&gt; John[4] \$gender [1] MALE Levels: FEMALE &lt; MALE</pre>
<pre>\$blood [1] O Levels: A B AB O</pre>	----->	<pre>&gt; John[5] \$blood [1] O Levels: A B AB O</pre>

R 코드 결과

# R의 데이터 구조

## ❖ 데이터 프레임 (data frame)

- 여러 개의 벡터가 모여서 만든 구조
- 벡터와 리스트의 측면을 둘 다 갖고 있음

이름 [1]	온도 [1]	감기상태 [1]	성별 [1]	혈액형 [1]
이름 [2]	온도 [2]	감기상태 [2]	성별 [2]	혈액형 [2]
이름 [3]	온도 [3]	감기상태 [3]	성별 [3]	혈액형 [3]

● DataFrame 데이터 구조



# R의 데이터 구조

## ❖ 데이터 프레임 (data frame)

속성값 (Attribute Value)

	subject_name	temperature	flu_status	gender	blood
1	John	32.2	FALSE	MALE	O
2	Dunpy	33.0	FALSE	FEMALE	AB
3	Steve	37.9	TRUE	MALE	A

예제  
(example)

# R의 데이터 구조

## ❖ 데이터 프레임 (data frame)

```
people <- data.frame(subject_name, temperature,  
                      flu_status, gender, blood,  
                      stringsAsFactors = FALSE)
```

● R 코드

	subject_name	temperature	flu_status	gender	blood
1	John	32.2	FALSE	MALE	O
2	Dunpy	33.0	FALSE	FEMALE	AB
3	Steve	37.9	TRUE	MALE	A

● R 코드 결과

# R의 데이터 구조

## ❖ 데이터 프레임 실습 1

```
people$subject_name
people[c("temperature", "flu_status")]
people[2:3] # 2번째 열과 3번째 열의 예제 출력
people[1,2] # 1번째 행과 2번째 열
people[c(1,3), c(2,4)] # 1,3번째 행/2,4번째 열
people[1, ] # 1행에 대한 모든 정보 출력
people[, 1] # 1열에 대한 모든 정보 출력
people[, ] # 모든 행과 열
```

- R 코드 (각각 행과 열을 어떻게 처리하는 지 보여주는 데이터 프레임 실습예제)

# R의 데이터 구조

## ❖ 데이터 프레임 실습 2

```
# 2행과 3, 5열 정보 빼기 (console)  
people[-2, c(-3, -5)]
```

```
# 2행과 3, 5열 정보 빼기 (view)  
view(people[-2, c(-3, -5)])
```

- R 코드 (각각 행과 열을 어떻게 처리하는 지 보여주는 데이터 프레임 실습예제)

# R의 데이터 구조

Vector

flu_status	logi [1:3]	FALSE	FALSE	TRUE
subject_name	chr [1:3]	"John"	"Dunpy"	"Steve"
temperature	num [1:3]	32.2	33	37.9

Factor

blood	Factor w/ 4 levels "A","B","AB","..."
gender	Factor w/ 2 levels "FEMALE","MALE..."

Scalar  
(Vector)



flu_status	FALSE
subject_name	"John Doe"
temperature	32.2

	subject_name	temperature	flu_status	gender	blood
1	John	32.2	FALSE	MALE	O
2	Dunpy	33.0	FALSE	FEMALE	AB
3	Steve	37.9	TRUE	MALE	A

Data	
John	List of 5
fullname	: chr "John"
temperature	: num 32.2
flu_status	: logi FALSE
gender	: Factor w/ 2 levels "FEMALE","MALE"
blood	: Factor w/ 4 levels "A","B","AB","..."

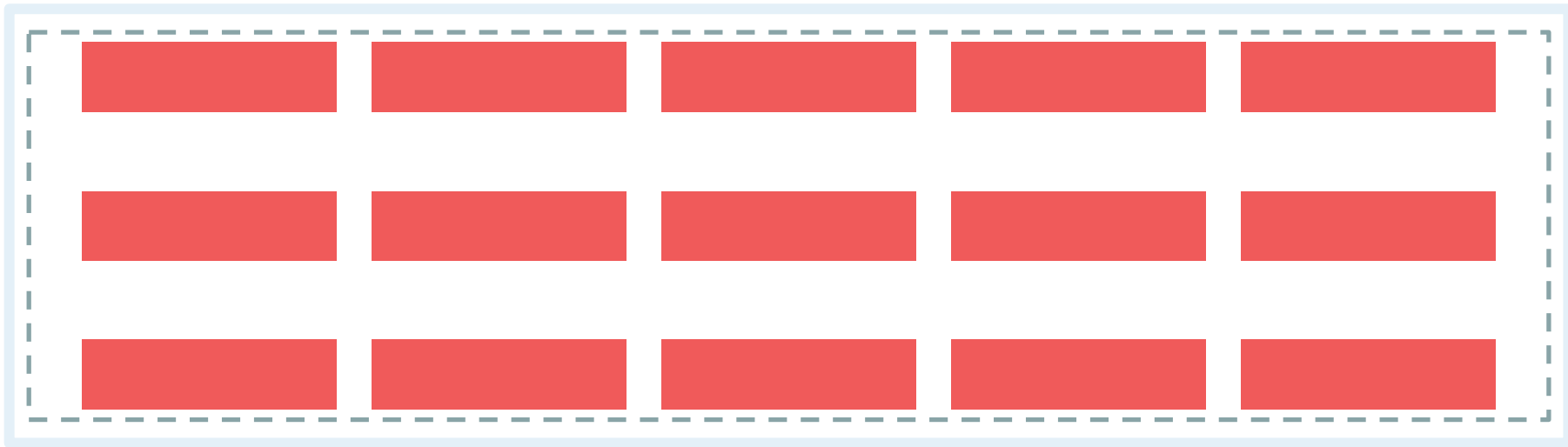
Data Frame

List

# R의 데이터 구조

## ❖ 매트릭스 (matrix)

- 데이터프레임과 달리 한가지 유형으로 2차원테이블 표현
- 수학적 연산에 주로 사용

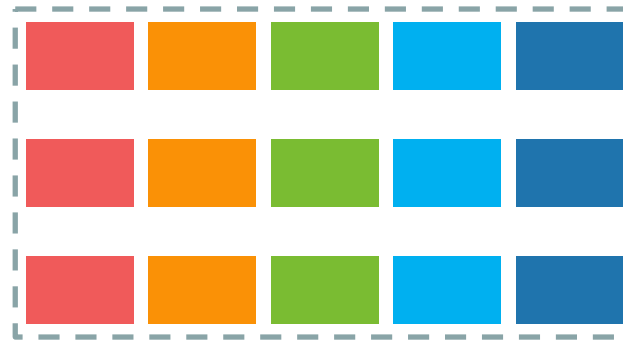


● Matrix 데이터 구조

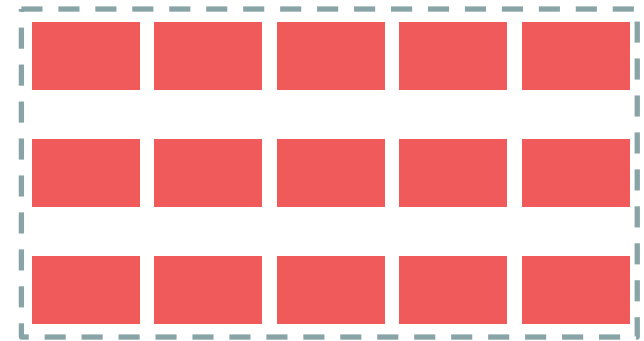
# R의 데이터 구조

## ❖ 데이터 구조

문자형	이름
실수형	온도
논리형	감기상태
팩터 (범주형/문자)	성별
팩터 (범주형/문자)	혈액형



Data Frame



Matrix

# R의 데이터 구조

## ❖ 매트릭스 실습

```
#nrow : 2행 나누기
m1 <- matrix(c('a','b','c','d'), nrow = 2)

#ncol : 2열 나누기
m2 <- matrix(c('a','b','c','d'), ncol = 2)

#nrow : 3행 나누기
m3 <- matrix(c('a','b','c','d','e','f'), nrow = 3)

#ncol : 3열 나누기
m4 <- matrix(c('a','b','c','d','e','f'), ncol = 3)

m4[1,1] # m4변수 1행 1열 출력
m4[2,3] # m4변수 2행 3열 출력

m3[1,]  # m3변수 1행 전체 출력
m3[,1]  # m3변수 1열 전체 출력
```

- R 코드 (각각 행과 열을 어떻게 처리하는 지 보여주는 실습예제)








# RData 저장하기 (R 작업상태 저장)

## ❖ RData 저장하고 로드하기

```
# 지금의 변수들을 저장  
save(m1,m2,m3,m4, file = "mydata.RData")  
  
# 데이터 변수 출력  
load("mydata.RData")
```

• R 코드

- ☐  그래프만들기.R
- ☐  그래프구조.R
- ☐  .RData
- ☐  .Rhstory
- ☐  mydata.RData

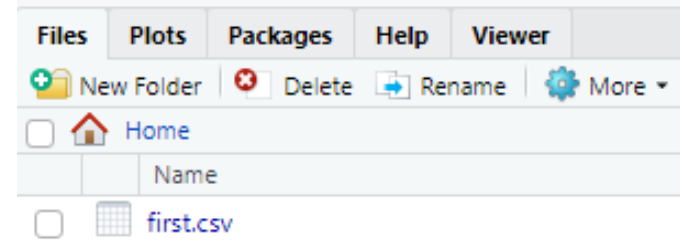
• R studio에 저장된 파일

# R 데이터를 CSV에 저장하고 불러오기

## ❖ people 데이터프레임을 "first.csv" 파일로 저장

```
write.csv(people, file = "first.csv")
```

• R 코드



## ❖ "first.csv" 파일을 first로 불러오기 ("file="은 생략 가능)

```
first <- read.csv(file = "first.csv")
```

• R 코드

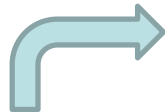
Data	
first	3 obs. of 6 variables
X :	int 1 2 3
subject_name :	Factor w/ 3 levels "Dunpy","John",...: 2 1 3
temperature :	num 32.2 33 37.9
flu_status :	logi FALSE FALSE TRUE
gender :	Factor w/ 2 levels "FEMALE","MALE": 2 1 2
blood :	Factor w/ 3 levels "A","AB","O": 3 2 1

# R 데이터를 CSV에 저장하고 불러오기

## ❖ CSV 파일을 factor 구조로 불러오지 않음

```
first <- read.csv("first.csv", stringsAsFactors = FALSE)
first <- first[, -1]
```

• R 코드



Data	
first	3 obs. of 6 variables
X : int 1 2 3	
subject_name:	chr "John" "Dunpy" "Steve"
temperature :	num 32.2 33 37.9
flu_status :	logi FALSE FALSE TRUE
gender :	chr "MALE" "FEMALE" "MALE"
blood :	chr "O" "AB" "A"

Data	
first	3 obs. of 6 variables
X : int 1 2 3	
subject_name:	Factor w/ 3 levels "Dunpy","John",...: 2 1 3
temperature :	num 32.2 33 37.9
flu_status :	logi FALSE FALSE TRUE
gender :	Factor w/ 2 levels "FEMALE","MALE": 2 1 2
blood :	Factor w/ 3 levels "A","AB","O": 3 2 1

# R 데이터를 CSV에 저장하고 불러오기

## ❖ read.csv의 header 비교

```
first1 <- read.csv("first.csv", stringsAsFactors = FALSE, header = TRUE)
first2 <- read.csv("first.csv", stringsAsFactors = FALSE, header = FALSE)
```

● R 코드

header = TRUE 인 경우

first	3 obs. of 6 variables
X : int 1 2 3	
subject_name: chr "John" "Dunpy" "Steve"	
temperature : num 32.2 33 37.9	
flu_status : logi FALSE FALSE TRUE	
gender : chr "MALE" "FEMALE" "MALE"	
blood : chr "O" "AB" "A"	

header = FALSE 인 경우

first	4 obs. of 6 variables
V1: int NA 1 2 3	
V2: chr "subject_name" "John" "Dunpy" "Steve"	
V3: chr "temperature" "32.2" "33" "37.9"	
V4: chr "flu_status" "FALSE" "FALSE" "TRUE"	
V5: chr "gender" "MALE" "FEMALE" "MALE"	
V6: chr "blood" "O" "AB" "A"	

# 데이터 이해와 탐구

## ❖ 데이터 구조 살펴보기

- str (변수명) : 데이터 프레임의 구조나 벡터, 리스트를 포함한 R 데이터 구조를 표시하는 방법

```
> usedcars <- read.csv("usedcars.csv", stringsAsFactors = FALSE)
> str(usedcars)
'data.frame': 150 obs. of 6 variables:
 $ year      : int  2011 2011 2011 2011 2012 2010 2011 2010 2011 2010 ...
 $ model     : chr   "SEL" "SEL" "SEL" "SEL" ...
 $ price     : int  21992 20995 19995 17809 17500 17495 17000 16995 16995 16995 ...
 $ mileage   : int  7413 10926 7351 11613 8367 25125 27393 21026 32655 36116 ...
 $ color     : chr   "Yellow" "Gray" "Silver" "Gray" ...
 $ transmission: chr   "AUTO" "AUTO" "AUTO" "AUTO" ...
```

- R 코드

# 수치 변수 살펴보기

## ❖ 요약 통계 값

- summary (변수명) : 일반적인 요약 통계를 보여주는 코드

```
> summary(usedcars$year)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 2000   2008   2009   2009   2010   2012

> summary(usedcars[c("price", "mileage")])
      price      mileage
Min.   : 3800  Min.   : 4867
1st Qu.:10995  1st Qu.: 27200
Median :13592  Median : 36385
Mean   :12962  Mean   : 44261
3rd Qu.:14904  3rd Qu.: 55125
Max.   :21992  Max.   :151479
```

- R 코드

# 중심 경향 측정 : 평균과 중앙값

## ❖ 평균과 중앙값

- 평균은 데이터의 중심을 측정하기 위해 사용되는 통계
- 중앙값은 값의 리스트에서 중앙에 있는 값 (median)

```
> (36000 + 44000 + 56000) / 3  
[1] 45333.33  
> mean(c(36000, 44000, 56000))  
[1] 45333.33
```

- R 코드 (평균)

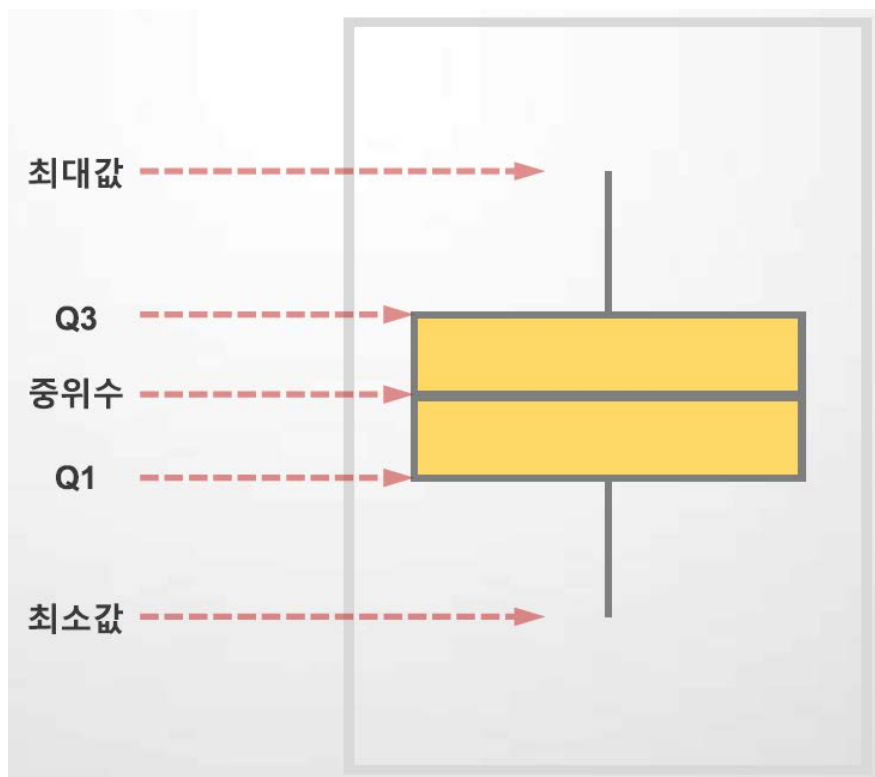
```
> median(c(36000, 44000, 56000))  
[1] 44000
```

- R 코드 (중앙값)

# 퍼짐 측정 : 사분위수와 다섯수치요약

## ❖ 사분위수 나타내기

- 범위(range)는 최대값과 최소값의 공간이라고 함
- range()와 차이 함수 diff()를 결합해 하나의 명령으로 데이터 범위 나타냄



```
> range(usedcars$price)  
[1] 3800 21992
```

- R 코드 (최소값과 최대값 범위 표현)

```
> diff(range(usedcars$price))  
[1] 18192
```

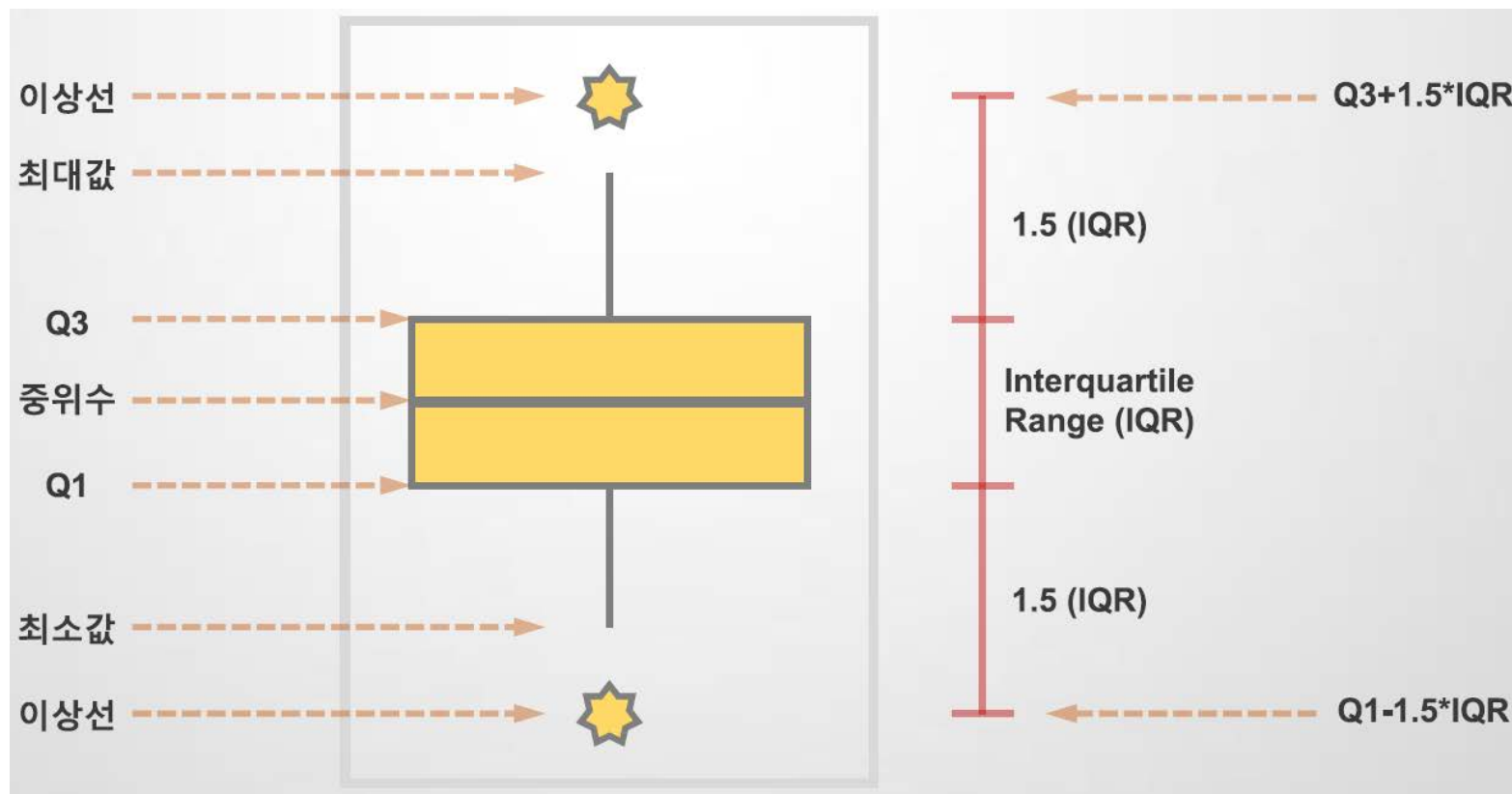
- R 코드 (range의 차이 값을 알아보기 위한 코드)



# 퍼짐 측정 : 사분위수와 다섯수치요약

## ❖ IQR 나타내기

- 1.5 IQR 이상인 Upper fence 위의 값을 보통 의심되는 이상값 혹은 극단값이라 함
- 보통 2.0 IQR 이상이면 \*로 표시하고, **보통  $IQR = Q3 - Q1$** 으로 3사분위수에서 1사분위수를 뺀 사분위수 범위 (Inter-Quartile Range : IQR)를 의미



# 퍼짐 측정 : 사분위수와 다섯수치요약

## ❖ IQR 나타내기

```
> IQR(usedcars$price)
[1] 3909.5
> quantile(usedcars$price)
      0%      25%      50%      75%     100%
3800.0 10995.0 13591.5 14904.5 21992.0
> quantile(usedcars$price, probs = c(0.01, 0.99))
      1%      99%
5428.69 20505.00
> quantile(usedcars$price, seq(from = 0, to = 1, by = 0.20))
      0%      20%      40%      60%      80%     100%
3800.0 10759.4 12993.8 13992.0 14999.0 21992.0
```

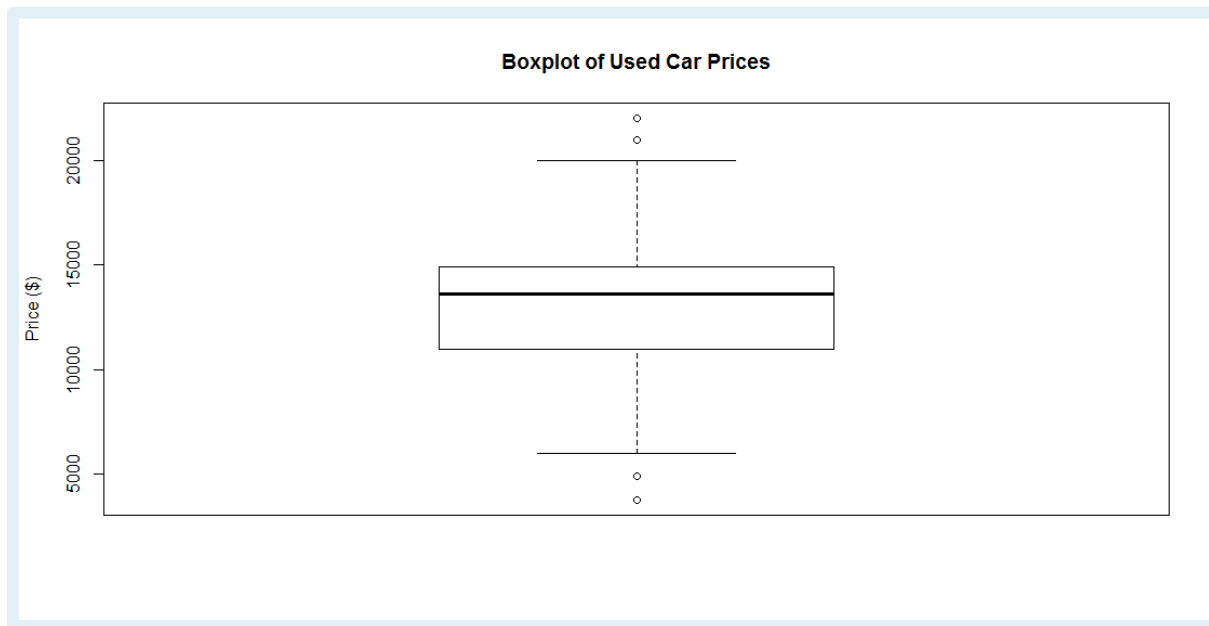
- R 코드 (IQR)

# 수치 변수 시각화 : boxplots

## ❖ Boxplots 1 (중고차의 가격 boxplots)

```
> boxplot(usedcars$price,  
+ main="Boxplot of Used Car Prices",  
+ ylab="Price ($)")
```

● R 코드



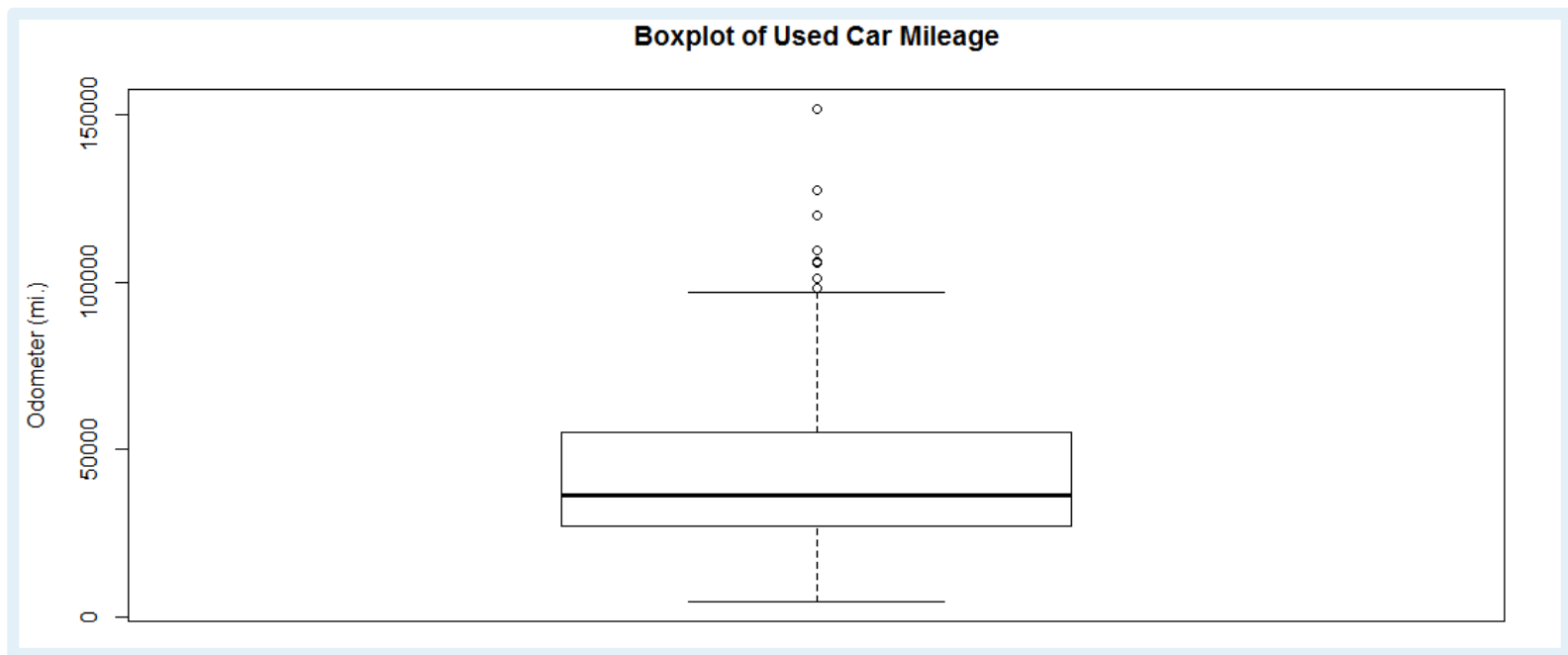
● R plots

# 수치 변수 시각화 : boxplots

## ❖ Boxplots 2 (중고차 주행거리에 대한 boxplots)

```
> boxplot(usedcars$mileage,  
+ main="Boxplot of Used Car Mileage",  
+ ylab="Odometer (mi.)")
```

● R 코드



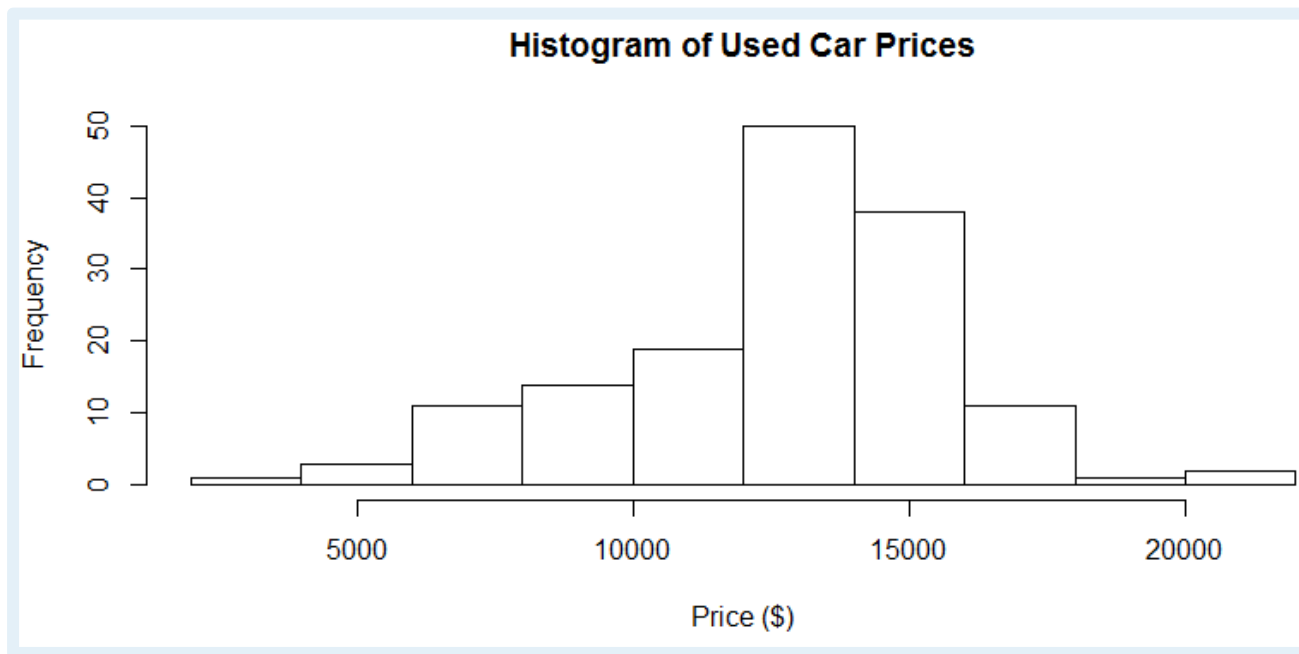
● R plots

# 수치 변수 시각화 : 히스토그램

## ❖ Histogram 1 (중고차 가격의 히스토그램)

```
> hist(usedcars$price,  
+ main = "Histogram of Used Car Prices",  
+ xlab = "Price ($)")
```

● R 코드



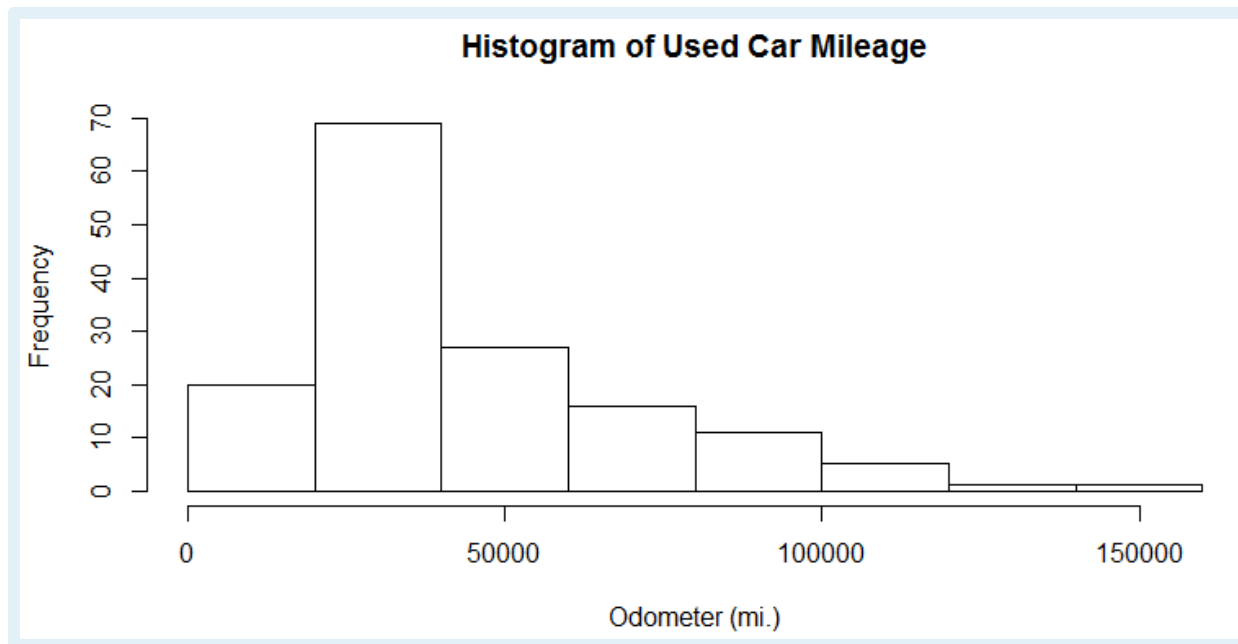
● R plots

# 수치 변수 시각화 : 히스토그램

## ❖ Histogram 2 (중고차 주행거리의 히스토그램)

```
> hist(usedcars$mileage,  
+ main = "Histogram of Used Car Mileage",  
+ xlab = "Odometer (mi.)")
```

● R 코드



● R plots

# 수치 데이터의 이해 : 분산과 표준 편차

## ❖ 분산과 표준편차

```
> var(usedcars$price)
[1] 9749892
> sd(usedcars$price)
[1] 3122.482
> var(usedcars$mileage)
[1] 728033954
> sd(usedcars$mileage)
[1] 26982.1
```

● R 코드

# 범주형 변수 살펴보기 : 일원배치

## ❖ 일원배치

```
> table(usedcars$year)

2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012
   3    1    1    1    3    2    6   11   14   42   49   16    1

> table(usedcars$model)

SE SEL SES
78  23  49

> table(usedcars$color)

Black  Blue  Gold  Gray  Green  Red Silver White Yellow
  35   17    1   16    5   25   32   16    3
```

- R 코드



# 범주형 변수 살펴보기

## ❖ 표의 비율 알아보기

```
> mt <- table(usedcars$model)
> prop.table(mt)
```

SE	SEL	SES
0.5200000	0.1533333	0.3266667

• R 코드

## ❖ 데이터 반올림

```
> rmt <- prop.table(mt)
> rmt <- prop.table(mt) * 100
> round(rmt, digits = 1)
```

SE	SEL	SES
52.0	15.3	32.7

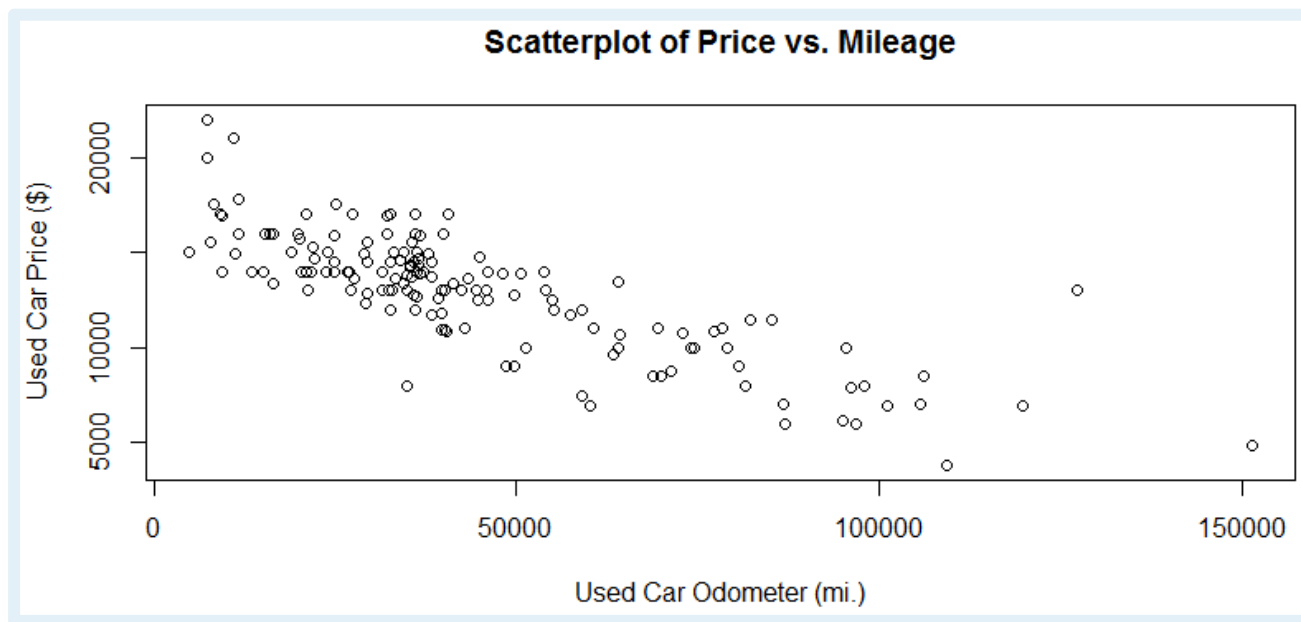
• R 코드

# 관계 시각화 : 산포도

## ❖ 관계 시각화 : 산포도

```
> plot(x = usedcars$mileage, y = usedcars$price,  
+ main = "Scatterplot of Price vs. Mileage",  
+ xlab = "Used Car Odometer (mi.)",  
+ ylab = "Used Car Price ($)")
```

● R 코드

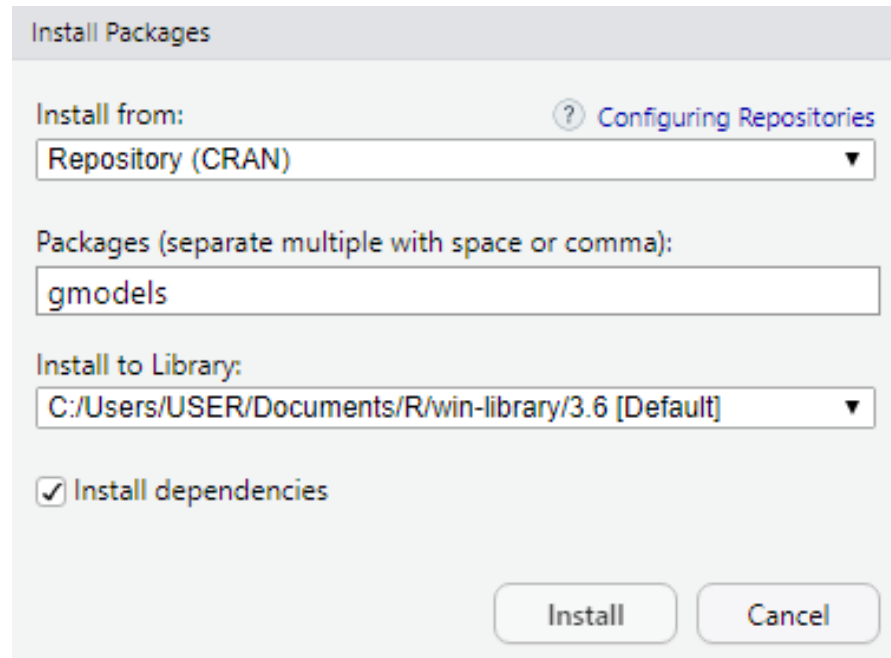


● R plots

# 이원배치표 패키지 설치

## ❖ 패키지 설치

- `install.packages("gmodels")` #gmodels 패키지 설치
- `library(gmodels)` #gmodels 불러오기



- R packages install

# 관계 살펴보기 : 이원배치표

## ❖ 변수 넣기

- 보수적인 "Black", "Gray", "Silver", "White " 색을 사용하는 사용자들만 나타내기 위해, 이 부분을 새로운 변수로 추가
- 150명 중 99명이 보수적인 차를 구매

```
> usedcars$conservative <-  
+ usedcars$color %in% c("Black", "Gray", "silver", "white")  
> table(usedcars$conservative)
```

```
FALSE  TRUE  
   51    99
```

- R 코드

# 관계 살펴보기 : 이원배치표

## ❖ 이원배치표 결과

```
> CrossTable(x = usedcars$model,
+ y = usedcars$conservative)
```

Cell Contents

	N
Chi-square contribution	
N / Row Total	
N / Col Total	
N / Table Total	

Total Observations in Table: 150

- R 코드 결과 - 1

usedcars\$model	usedcars\$conservative		Row Total
	FALSE	TRUE	
SE	27	51	78
	0.009	0.004	
	0.346	0.654	0.520
	0.529	0.515	
SEL	0.180	0.340	
	7	16	23
	0.086	0.044	
	0.304	0.696	0.153
SES	0.137	0.162	
	0.047	0.107	
	17	32	49
	0.007	0.004	
Column Total	0.347	0.653	0.327
	0.333	0.323	
	0.113	0.213	
	51	99	150
	0.340	0.660	

- R 코드 결과 - 2

# 함수 만들기

- ❖ 함수: 하나 이상의 명령어로 반복 사용 가능한 기능을 구현한 것

