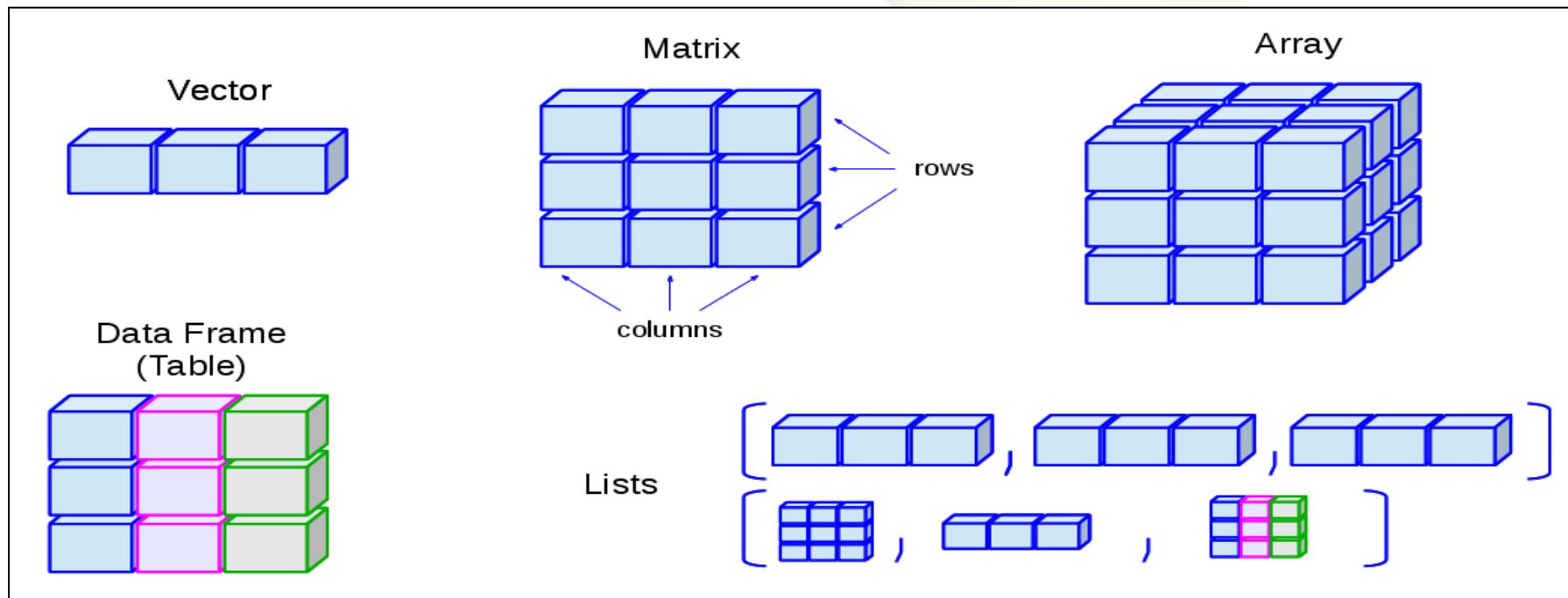


# **Complex Data Structure**

# R 복합 데이터 구조

이름	차원	개별 데이터 구조	포함하는 데이터
벡터 (Vector)	1차원	값으로만 구성	단일 형식의 데이터로 구성
행렬 (Matrix)	2차원	값으로만 구성	
배열 (Array)	n차원	값으로만 구성	
리스트 (List)	n차원	Key-Value 형식	하나 이상의 데이터 형식으로 구성
데이터프레임 (DataFrame)	2차원	Key-Value 형식	



# R 벡터

- 여러 개의 동일한 자료형 데이터를 모아서 저장 관리하는 타입
- 다른 자료형이 지정될 경우 강제 형변환을 통해 일치 시킴
- `c(데이터목록)` 함수를 사용해서 생성

```
> c(1, 2, 3, 4, 5)
[1] 1 2 3 4 5

> c(1, 2, 3, 4, "5")
[1] "1" "2" "3" "4" "5"

> vec1 <- c(1, 2, 3, 4, 5); vec1;
[1] 1 2 3 4 5
```

# R 벡터 요소 접근

- 1부터 시작하는 위치 값을 이용해서 접근
- 위치 값이 음수인 경우 해당 위치 값을 제외한 요소 접근
- 인덱스의 목록 또는 범위를 이용해서 여러 요소 접근 가능

```
> vec1  
[1] 1 2 3 4 5  
> vec1[3]  
[1] 3  
> vec1[-3]  
[1] 1 2 4 5  
> vec1[1:(length(vec1) - 2)]  
[1] 1 2 3  
> vec1[-1:-3]  
[1] 4 5  
> vec1[2:4]  
[1] 2 3 4  
> vec1[2] <- 6  
> vec1  
[1] 1 6 3 4 5
```

```
> vec1 <- c(vec1, 7)  
> vec1  
[1] 1 6 3 4 5 7  
  
> vec1[9] <- 9  
> vec1  
[1] 1 6 3 4 5 7 NA NA 9  
  
> append(vec1, 10, after = 3)  
[1] 1 6 3 10 4 5 7 NA NA 9  
> append(vec1, c(10, 11), after = 3)  
[1] 1 6 3 10 11 4 5 7 NA NA 9  
> append(vec1, 12, after = 0)  
[1] 12 1 6 3 4 5 7 NA NA 9
```

# R 벡터 연산

```
> var1 <- c(1, 2, 3)
> var2 <- c(3, 4, 5)
> var1 + var2 # 요소별 데이터 합
[1] 4 6 8
> var3 <- c('3', '4', 5)
> var1 + var3
Error in var1 + var3 : 이항연산자에 수치가 아닌 인수입니다
> union(var1, var3) # 두 벡터 결합 (중복 데이터 제거)
[1] "1" "2" "3" "4" "5"
> var4 <- c(1, 2, 3, 4, 5)
> var1 + var4
[1] 2 4 6 5 7
경고메시지(들):
In var1 + var4 : 두 객체의 길이가 서로 배수관계에 있지 않습니다
> var1 - var2
[1] -2 -2 -2
> setdiff(var1, var2) # 차집합
[1] 1 2
> setdiff(var2, var1) # 차집합
[1] 4 5
> intersect(var1, var2) # 교집합
[1] 3
```

# R 벡터

- 벡터의 각 요소에 이름 지정

```
> fruits <- c(10, 20, 30)
> fruits
[1] 10 20 30
> names(fruits) <- c("apple", "banana", "peach")
> fruits
apple banana peach
 10      20      30
```

- 벡터에 연속적인 데이터 할당 → seq(), rep()

```
> var5 <- seq(1, 5); var5
[1] 1 2 3 4 5
> var6 <- seq(2, -2); var6
[1] 2 1 0 -1 -2
> var7 <- seq(1, 10, 2); var7
[1] 1 3 5 7 9
```

```
> var8 <- rep(1:3, 2); var8;
[1] 1 2 3 1 2 3
> var9 <- rep(1:3, each = 2); var9
[1] 1 1 2 2 3 3
```

# R 벡터

- 벡터의 길이 찾기 → NROW() 함수 사용
  - 참고 : Matrix, Array, DataFrame 의 경우 nrow() 함수도 사용 가능

```
> var1  
[1] 1 2 3  
> length(var1)  
[1] 3  
> NROW(var1)  
[1] 3
```

- 벡터에 특정 데이터 포함 여부 확인 → %in% 연산자 사용

```
> var7  
[1] 1 3 5 7 9  
> 3 %in% var7  
[1] TRUE  
> 4 %in% var7  
[1] FALSE
```

# R 행렬 (Matrix)

- 행과 열로 구성된 데이터 형식 (벡터의 집합)
- 동일한 자료형의 데이터로 구성됨
- `matrix()` 함수로 생성

```
> mat1 <- matrix(c(1, 2, 3, 4)); mat1
      [,1]
[1,]    1
[2,]    2
[3,]    3
[4,]    4

> mat2 <- matrix(c(1, 2, 3, 4), nrow = 2); mat2
      [,1] [,2]
[1,]    1    3
[2,]    2    4

> mat3 <- matrix(c(1, 2, 3, 4), ncol = 2); mat3
      [,1] [,2]
[1,]    1    3
[2,]    2    4
```

# R 행렬 내부의 데이터 접근

- 행과 열의 인덱스를 ,로 나열해서 표시
- 나머지 규칙은 벡터와 동일

```
> mat3  
      [,1] [,2]  
[1,]    1    3  
[2,]    2    4  
  
> mat3[,1]  
[1] 1 2  
  
> mat3[1,]  
[1] 1 3  
  
> mat3[1,1]  
[1] 1
```

```
> mat3[-1,2]  
[1] 4  
  
> mat3[,-1]  
[1] 3 4  
  
> mat3[-1,]  
[1] 2 4  
  
> mat3[1:2,2]  
[1] 3 4
```

# R 행렬

- 새로운 행 추가 → rbind(), cbind() 사용

```
> mat4 <- matrix(c(1,2,3,4,5,6,7,8,9), nrow = 3, byrow = T)
> mat4
  [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9

> mat4 <- rbind(mat4, c(11, 12, 13))
> mat4
  [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
[4,]   11   12   13
```

# R 행렬

- 새로운 행 추가 → rbind(), cbind() 사용

```
> mat4 <- rbind(mat4, c(15, 16, 17, 18))
Warning message:
In rbind(mat4, c(15, 16, 17, 18)) :
  number of columns of result is not a multiple of vector length (arg 2)
> mat4
```

	[,1]	[,2]	[,3]
[1,]	1	2	3
[2,]	4	5	6
[3,]	7	8	9
[4,]	11	12	13
[5,]	15	16	17

추가되는 벡터가 크면 남는 부분은 버림

```
> mat4 <- rbind(mat4, c(19, 20))
Warning message:
In rbind(mat4, c(19, 20)) :
  number of columns of result is not a multiple of vector length (arg 2)
> mat4
```

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

추가되는 벡터가 작으면 기존 데이터 재사용(Recycling)

# R 행렬

- 새로운 열 추가 → cbind() 사용

```
> mat5 <- matrix(c('a', 'b', 'c', 'd'), nrow = 2, byrow = T)
> mat5
[,1] [,2]
[1,] "a"   "b"
[2,] "c"   "d"
> mat5 <- cbind(mat5, c('e', 'f'))
> mat5
[,1] [,2] [,3]
[1,] "a"   "b"   "e"
[2,] "c"   "d"   "f"
```

- 행과 열의 이름 지정

```
> colnames(mat5) <- c("First", "Second", "Third")
> rownames(mat5) <- c("row1", "row2")
> # dimnames(mat5) <- list(c("row1","row2"),c("First","Second","Third"))
> mat5
      First Second Third
row1 "a"     "b"     "e"
row2 "c"     "d"     "f"
```

# R 행렬

## ▪ 행렬 연산 (스칼라 연산)

```
> x <- matrix(1:9, nrow=3)
> x * 2
[,1] [,2] [,3]
[1,]    2     8    14
[2,]    4    10    16
[3,]    6    12    18
> x / 2
[,1] [,2] [,3]
[1,]  0.5   2.0   3.5
[2,]  1.0   2.5   4.0
[3,]  1.5   3.0   4.5
> x + 3
[,1] [,2] [,3]
[1,]    4     7    10
[2,]    5     8    11
[3,]    6     9    12
> x - 2
[,1] [,2] [,3]
[1,]   -1     2     5
[2,]    0     3     6
[3,]    1     4     7
```

## ▪ 행렬 연산 (행렬간 덧셈, 뺄셈)

```
> x <- matrix(1:9, nrow = 3)
> x
[,1] [,2] [,3]
[1,]    1     4     7
[2,]    2     5     8
[3,]    3     6     9
> x + x
[,1] [,2] [,3]
[1,]    2     8    14
[2,]    4    10    16
[3,]    6    12    18
> x - x
[,1] [,2] [,3]
[1,]    0     0     0
[2,]    0     0     0
[3,]    0     0     0
```

# R 행렬

## ▪ 행렬 연산 (행렬간 곱셈)

```
> x <- matrix(1:9, nrow = 3)
> x
      [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9

> x %*% x
      [,1] [,2] [,3]
[1,]   30   66  102
[2,]   36   81  126
[3,]   42   96  150
```

## ▪ 행렬 연산 (역행렬)

```
> x <- matrix(1:4, ncol = 2)
> x
      [,1] [,2]
[1,]    1    3
[2,]    2    4

> solve(x)
      [,1] [,2]
[1,]   -2   1.5
[2,]    1  -0.5

> x %*% solve(x) # 단위 행렬 만들기
      [,1] [,2]
[1,]    1    0
[2,]    0    1
```

# R 행렬

## ▪ 행렬 연산 (전치 행렬)

```
> x <- matrix(1:9, nrow = 3)
> x
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
> t(x)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

## ▪ 행렬의 차원(행 / 열 개수)

```
> nrow(x)
[1] 3
> ncol(x)
[1] 3
```

# R 배열

- 2차원 이상 n차원 데이터 저장 형식 (행렬의 집합)
  - 동일한 데이터 형식
  - array() 함수 사용해서 생성

```
> array1 <- array(c(1:12), dim = c(4, 3))
> array1
[,1] [,2] [,3]
[1,]    1    5    9
[2,]    2    6   10
[3,]    3    7   11
[4,]    4    8   12
```

# R 배열

- 3차원 데이터 저장 형식 (행렬의 집합) 계속

```
> array2 <- array(c(1:12), dim = c(2, 2, 3))
> array2
, , 1

[,1] [,2]
[1,]    1    3
[2,]    2    4

, , 2

[,1] [,2]
[1,]    5    7
[2,]    6    8

, , 3

[,1] [,2]
[1,]    9   11
[2,]   10   12
```

# R 배열 요소 접근

- 각 차원의 인덱스를 이용
  - 일반적인 규칙은 벡터, 행렬과 동일

```
> array2[1,1,3]
[1] 9
> array2[,1,3]
[1] 9 10
> array2[1,,]
     [,1] [,2] [,3]
[1,]    1    5    9
[2,]    3    7   11
> array2[,2,]
     [,1] [,2] [,3]
[1,]    3    7   11
[2,]    4    8   12
> array2[,-2,]
     [,1] [,2] [,3]
[1,]    1    5    9
[2,]    2    6   10
```

# R 리스트

- 키 - 값 형식의 데이터 집합

- 값으로 저장되는 데이터는 스칼라 및 집합 형식 데이터 가능
- 값에는 요소별로 각각 다른 형식의 데이터 저장 가능
- `list()` 함수로 생성

```
> list1 <- list(name = 'John Doe', address = 'Seoul, Korea', phone =
  '010-0000-0000', salary = 1000)
> list1
$name
[1] "John Doe"

$address
[1] "Seoul, Korea"

$phone
[1] "010-0000-0000"

$salary
[1] 1000
```

# R 리스트 요소 접근

- 리스트 이름 \$ 키 이름 또는 요소의 인덱스로 접근

```
> list1$name  
[1] "John Doe"  
  
> list1[1:2]  
$name  
[1] "John Doe"  
  
$address  
[1] "Seoul, Korea"  
  
> list1[3]  
$phone  
[1] "010-0000-0000"  
  
> list1[c(1, 3)]  
$name  
[1] "John Doe"  
  
$phone  
[1] "010-0000-0000"
```

인덱스로 값에 직접 접근 할 때는  
[[index]] 사용

```
> list1[[1]]  
[1] "John Doe"  
> list1[[2]]  
[1] "Seoul, Korea"  
> list1[[3]]  
[1] "010-0000-0000"
```

# R 리스트 요소 추가 및 삭제

```
> list1$birth <- '1980-5-7'  
> list1$name <- c("John Doe",  
                 "Johnny")  
  
> list1  
$name  
[1] "John Doe" "Johnny"  
  
$address  
[1] "Seoul, Korea"  
  
$phone  
[1] "010-0000-0000"  
  
$salary  
[1] 1000  
  
$birth  
[1] "1980-5-7"
```

```
> list1$birth <- NULL  
> list1  
$name  
[1] "John Doe" "Johnny"  
  
$address  
[1] "Seoul, Korea"  
  
$phone  
[1] "010-0000-0000"  
  
$salary  
[1] 1000
```

# R 데이터 프레임

- 테이블 구조의 데이터 집합
  - 표(table) 형식의 데이터
  - 열 단위 스키마 형성 → 열마다 서로 다른 자료형의 데이터 저장 가능
  - 열은 데이터의 속성으로 열의 이름을 지정할 수 있음
  - 행은 한 건의 데이터를 의미
  - `data.frame` 함수로 생성

열 (Column)

행 (Row)

성별	연령	학점	연봉 (만원)
남자	26	3.8	2700
여자	42	4.2	4000
남자	35	2.6	3500

# R 데이터 프레임

- 벡터로부터 데이터프레임 생성

```
> no <- c(1, 2, 3, 4)
> name <- c("Apple", "Peach", "Banana", "Grape")
> price <- c(500, 200, 100, 50)
> qty <- c(5, 2, 4, 7)

> sales <- data.frame(NO = no, NAME = name, PRICE = price, QTY = qty)
> sales
  NO    NAME PRICE QTY
1  1   Apple   500    5
2  2   Peach   200    2
3  3 Banana   100    4
4  4   Grape    50    7
```

# R 데이터 프레임

## ▪ 행렬로부터 데이터프레임 생성

- 데이터프레임 생성 시 자료형 변환에 주의 (아래는 모두 문자열로 변경)

```
> sales2 <- matrix(c(1, 'Apple', 500, 5,
+                      2, 'Peach', 200, 2,
+                      3, 'Banana', 100, 4,
+                      4, 'Grape', 50, 7), nrow = 4, byrow = T)
> sales2
     [,1] [,2]    [,3]    [,4]
[1,] "1"  "Apple" "500"  "5"
[2,] "2"  "Peach" "200"  "2"
[3,] "3"  "Banana" "100"  "4"
[4,] "4"  "Grape" "50"   "7"

> df1 <- data.frame(sales2)
> df1
  X1     X2    X3 X4
1 1 Apple 500  5
2 2 Peach 200  2
3 3 Banana 100  4
4 4 Grape 50   7
```

# R 데이터 프레임 데이터 접근

- 데이터프레임이름\$컬럼이름 또는 행, 열의 인덱스로 데이터 접근

```
> sales
  NO   NAME PRICE QTY
1 1 Apple   500   5
2 2 Peach   200   2
3 3 Banana  100   4
4 4 Grape    50    7
```

```
> sales$NAME
[1] Apple  Peach  Banana Grape
Levels: Apple Banana Grape Peach
```

```
> sales[1, 3]
[1] 500
```

```
> sales[1, ]
  NO   NAME PRICE QTY
1 1 Apple   500   5
```

```
> sales[,1]
[1] 1 2 3 4
```

# R 데이터 프레임 데이터 접근

- 행, 열의 인덱스로 데이터 접근할 때 벡터, 행렬 등의 규칙과 동일

```
> sales[c(1,2),]  
NO NAME PRICE QTY  
1 1 Apple 500 5  
2 2 Peach 200 2
```

```
> sales[,c(1,2)]  
NO NAME  
1 1 Apple  
2 2 Peach  
3 3 Banana  
4 4 Grape
```

```
> sales[,c(1:3)]  
NO NAME PRICE  
1 1 Apple 500  
2 2 Peach 200  
3 3 Banana 100  
4 4 Grape 50
```

```
> sales[-1,]  
NO NAME PRICE QTY  
2 2 Peach 200 2  
3 3 Banana 100 4  
4 4 Grape 50 7
```

```
> sales[, -1]  
NAME PRICE QTY  
1 Apple 500 5  
2 Peach 200 2  
3 Banana 100 4  
4 Grape 50 7
```

# R 데이터 프레임 데이터 접근

- 한 개의 컬럼만 선택할 경우 결과가 벡터로 반환됨 → 피하려면 drop 사용

```
> d <- data.frame(x = c(1, 2, 3, 4, 5), y = c(2, 4, 6, 8, 10))
> d
  x  y
1 1  2
2 2  4
3 3  6
4 4  8
5 5 10

> d[,c("x")]
[1] 1 2 3 4 5
> d[,c("x"), drop = F]
  x
1 1
2 2
3 3
4 4
5 5
```

# R 데이터 프레임 사용

- 부분 데이터 추출 → subset() 함수 사용

```
> subset(sales, qty < 5)
NO   NAME PRICE QTY
2  2 Peach   200    2
3  3 Banana  100    4

> subset(sales, price == 200)
NO   NAME PRICE QTY
2  2 Peach   200    2

> subset(sales, name == 'Apple')
NO   NAME PRICE QTY
1  1 Apple   500    5
```

# R 데이터 프레임 사용

- 행과 열의 개수가 같은 데이터프레임 병합 → rbind(), cbind() 함수 사용

```
> no <- c(1, 2, 3)
> name <- c('apple', 'banana', 'peach')
> price <- c(100, 200, 300)
> df1 <- data.frame(NO = no, NAME = name, PRICE = price)
> df1
  NO    NAME PRICE
1 1    apple   100
2 2  banana   200
3 3    peach   300

> no <- c(10, 20, 30)
> name <- c("train", "car", "airplane")
> price <- c(1000, 2000, 3000)
> df2 <- data.frame(NO = no, NAME = name, PRICE = price)
> df2
  NO    NAME PRICE
1 10    train  1000
2 20      car  2000
3 30  airplane 3000
```

# R 데이터 프레임 사용

- 행 또는 열 방향으로 데이터프레임 병합 → rbind(), cbind() 함수 사용
  - 행 방향으로 병합하는 경우 속성의 개수와 이름이 동일할 때 가능

```
> df3 <- rbind(df1, df2)
> df3
  NO      NAME PRICE
1 1     apple   100
2 2    banana   200
3 3     peach   300
4 10    train  1000
5 20      car  2000
6 30 airplane 3000

> df4 <- cbind(df1, df2)
> df4
  NO      NAME PRICE NO      NAME PRICE
1 1     apple   100 10    train  1000
2 2    banana   200 20      car  2000
3 3     peach   300 30 airplane 3000
```

# R 데이터 프레임 사용

- 데이터 프레임에 열 및 행 추가

```
> df1
  name price
1 apple    300
2 banana   200
3 cherry   100
> df1 <- rbind(df1, data.frame(name = 'berry', price = 500))
> df1
  name price
1 apple    300
2 banana   200
3 cherry   100
4 berry    500
> df1 <- cbind(df1, data.frame(qty = c(10, 20, 30, 40)))
> df1
  name price qty
1 apple    300   10
2 banana   200   20
3 cherry   100   30
4 berry    500   40
```

# R 데이터 프레임 사용

- 특정 컬럼을 기준으로 데이터프레임 병합 → `merge()` 함수 사용

```
> df1 <- data.frame(name = c('apple', 'banana', 'cherry'),  
                      price = c(300, 200, 100))  
> df2 <- data.frame(name = c('apple', 'cherry', 'berry'),  
                      qty = c(10, 20, 30))
```

```
> df1  
  name price  
1 apple    300  
2 banana   200  
3 cherry   100
```

```
> df2  
  name qty  
1 apple  10  
2 cherry 20  
3 berry  30
```

```
> merge(df1, df2)  
  name price qty  
1 apple    300  10  
2 cherry   100  20  
> merge(df1, df2, all = T)  
  name price qty  
1 apple    300  10  
2 banana   200  NA  
3 cherry   100  20  
4 berry     NA  30  
> cbind(df1, df2)  
  name price  name qty  
1 apple    300  apple  10  
2 banana   200  cherry 20  
3 cherry   100  berry  30
```

# R 데이터 프레임 사용

- 데이터 프레임에서 특정 컬럼 추출

```
> no <- c(1, 2, 3, 4, 5)
> name <- c("장동건", "김윤석", "송강호", "최민식")
> address <- c("서울", "대전", "포항", "경주")
> phone <- c(1111, 2222, 3333, 4444, 5555)
> hobby <- c("독서", "미술", "낚시", "등산")
> member <- data.frame(NO = no, NAME = name, ADDRESS = address,
                           PHONE = phone, HOBBY = hobby)

> member
   NO   NAME ADDRESS PHONE      HOBBY
1  1 장동건    서울   1111      독서
2  2 김윤석    대전   2222      미술
3  3 송강호    포항   3333      낚시
4  4 최민식    경주   4444      등산
```

# R 데이터 프레임 사용

## ■ 데이터 프레임에 특정 컬럼 추출 (계속)

```
> member2 <- subset(member, select = c(NO, NAME, PHONE)); member2;
NO   NAME PHONE
1 1 장동건 1111
2 2 김윤석 2222
3 3 송강호 3333
4 4 최민식 4444
> member3 <- subset(member, select = -PHONE); member3;
NO   NAME ADDRESS      HOBBY
1 1 장동건    서울      독서
2 2 김윤석    대전      미술
3 3 송강호    포항      낚시
4 4 최민식    경주      등산
> colnames(member3) <- c("번호", "이름", "주소", "취미"); member3;
번호   이름 주소      취미
1 1 장동건 서울      독서
2 2 김윤석 대전      미술
3 3 송강호 포항      낚시
4 4 최민식 경주      등산
```

# R 데이터 프레임 사용

## ▪ 기타 데이터 프레임에 사용하는 함수

함수	설명
<code>ncol(dataframe)</code>	dataframe의 열의 개수
<code>nrow(dataframe)</code>	dataframe의 행의 개수
<code>names(dataframe)</code>	dataframe의 열이름 출력
<code>rownames(dataframe)</code> <code>row.names(dataframe)</code>	dataframe의 행이름 출력
<code>colnames(dataframe)</code>	dataframe의 열이름 출력
<code>dataframe[c(1, 2, 3, 4, 5), ]</code>	1, 2, 3, 4, 5 행의 순서대로 출력
<code>dataframe[c(1, 2, 3, 4, 5)]</code>	1, 2, 3, 4, 5 열의 순서대로 출력

# R 데이터 프레임

- 기타 데이터프레임에 사용하는 함수 (계속)

```
> sales
  NO   NAME PRICE QTY
1 1 Apple  500    5
2 2 Peach  200    2
3 3 Banana 100    4
4 4 Grape   50    7
> ncol(sales)
[1] 4
> nrow(sales)
[1] 4
> names(sales)
[1] "NO"     "NAME"   "PRICE"  "QTY"
> rownames(sales)
[1] "1" "2" "3" "4"
> sales[c(2, 3, 1),]
  NO   NAME PRICE QTY
2 2 Peach  200    2
3 3 Banana 100    4
1 1 Apple  500    5
```