Outline

day 2 – March 20th

- Missing values
- Two-dimensional data structures:
  - Matrix
  - Data Frames

NA: Not Available

- **NA** (not available) is a recognized element in R.

- Finding missing values in a vector

```
x <- c(4, 2, 7, NA)
is.na(x)
[1] FALSE FALSE FALSE TRUE
```

NA: Not Available

- x <- c(4, 2, 7, NA)

- Removing missing values

  x <- na.omit(x)

  x <- x[!is.na(x)]

- x < 3

[1] FALSE TRUE FALSE NA

NA: Not Available

- Functions can deal with NAs and/or have arguments to deal with NAs:

```
x <- c(4, 2, 7, NA)
mean(x)
[1] NA
mean(x, na.rm=TRUE)
[1] 4.333333
```

Data Structures

Vectors

Matrices

Data frames

Matrices

- A matrix is a vector of **2 dimensions**

- All columns in a matrix must have :
  - the same **type** (numeric, character, logical)
  - the same **length**

Matrices

- Create a matrix **from vectors with the rbind function:**

```
x <- c(1, 44); y <- c(0, 12); z <- c(34, 4)
b <- rbind(x, y, z)
```

- Create a matrix **with the matrix function**:

```
b <- matrix(c(1, 0, 34, 44, 12, 4),
            nrow=3,
            ncol=2)
```

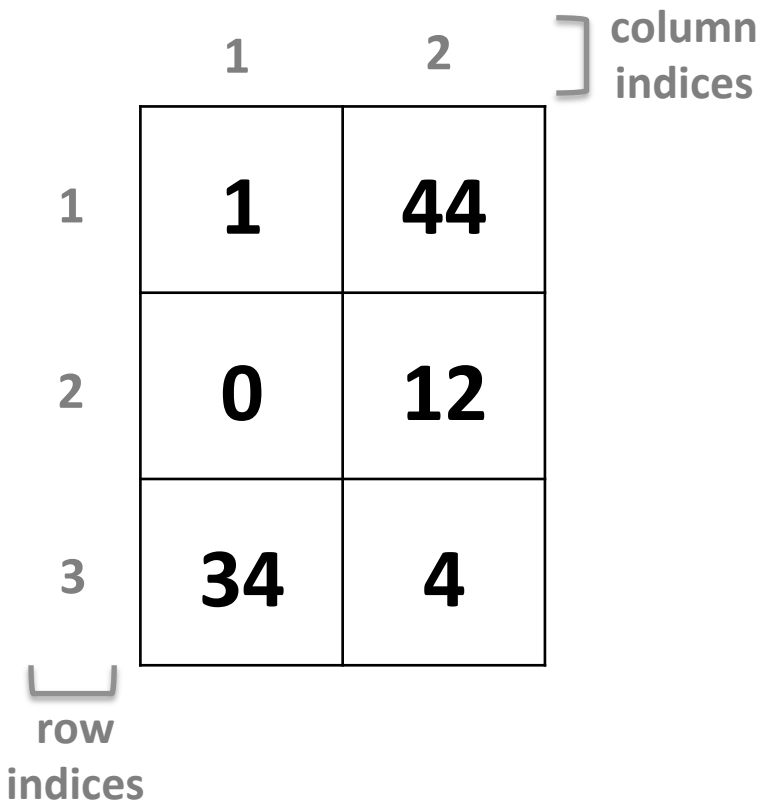
Matrices

b

1	**44**
0	**12**
34	**4**

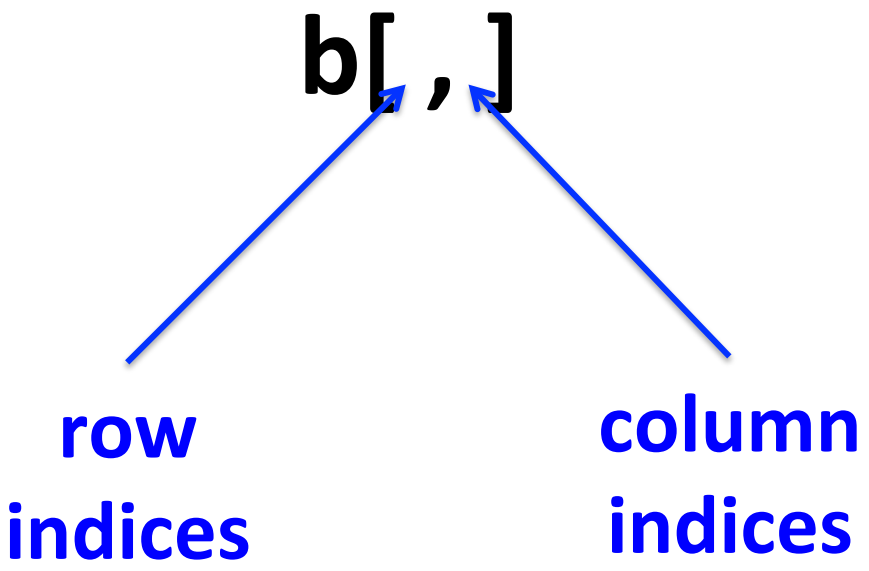
Matrices

b



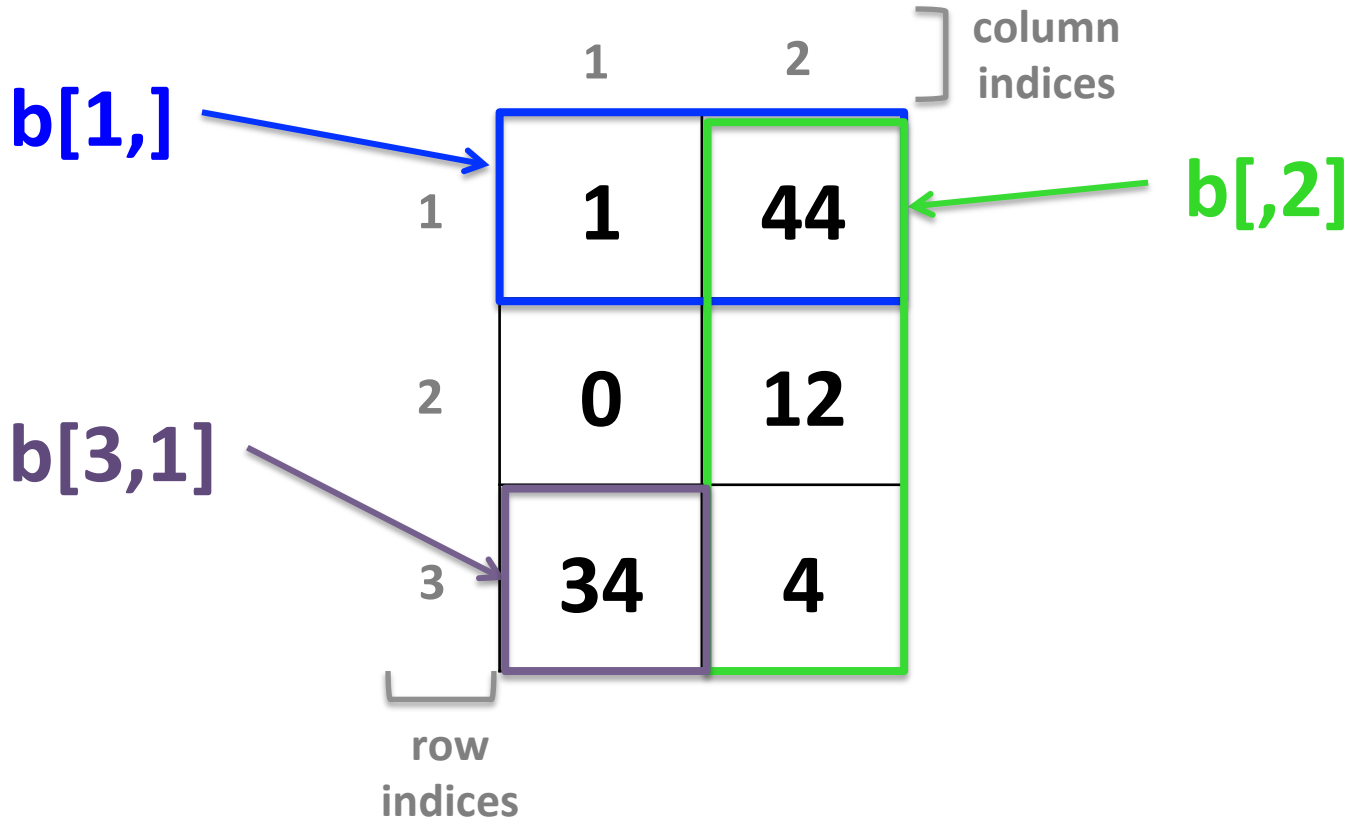
Matrices

- Fetch rows, columns or single elements of a matrix:



Matrices

b



Manipulate matrices

- **add 1** to all elements of a matrix

```
b <- b + 1
```

- **multiply by 3** all elements of a matrix

```
b <- b * 3
```

- **subtract 2** to each element of the first row of a matrix

```
b[1, ] <- b[1, ] – 2
```

- **replace** elements that comply a condition:

```
b[ b>3 ] <- 4
```

Data frames

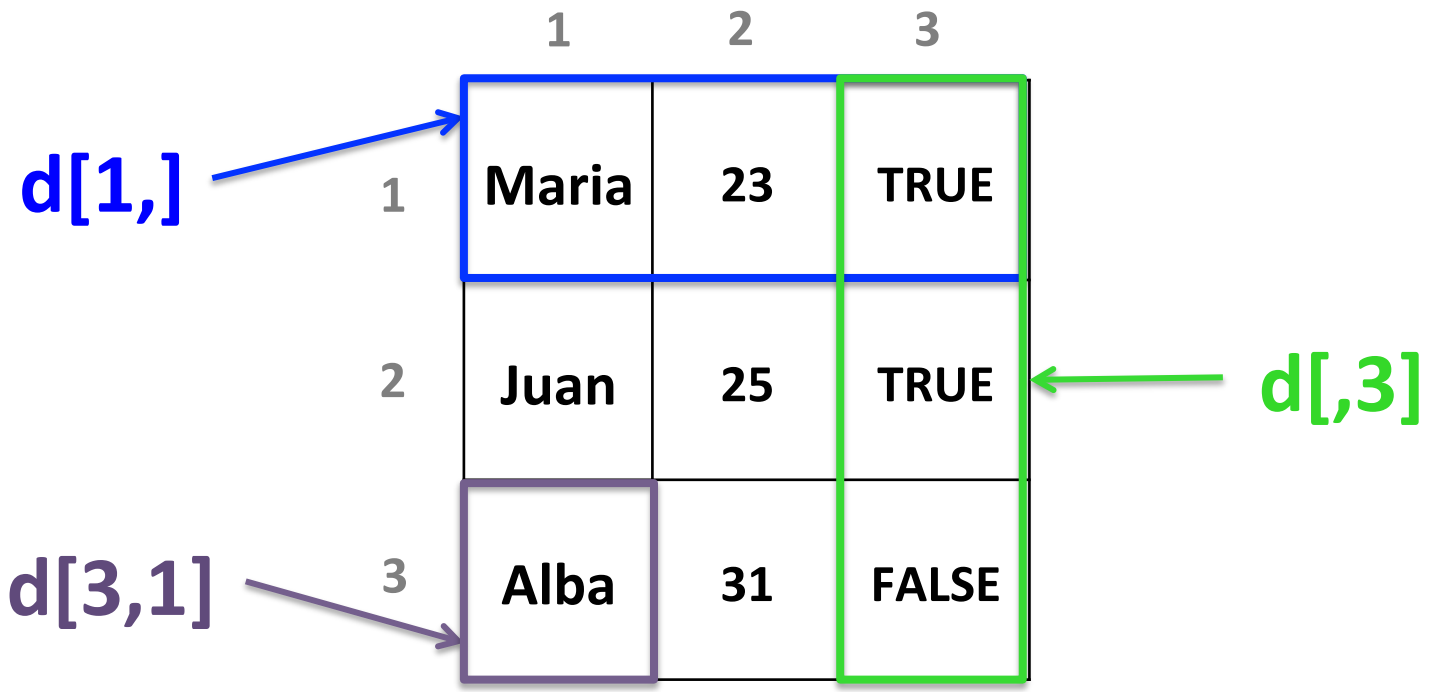
- Structures of 2 dimensions.
- More general than matrices.
- Different columns can have **different types** but must have the **same length.**

```
d <- data.frame(c("Maria", "Juan", "Alba"),
                c(23, 25, 31),
                c(TRUE, TRUE, FALSE),
                stringsAsFactors = FALSE)
```

Data frames

- Fetch rows, columns or single elements of a data frame:

d



Dimensions

- Get dimension of data frame or matrix

- **dim**(b)

  first element: number of rows

  second element: number of columns
- **nrow**(b)
- **ncol**(b)

Dimension names

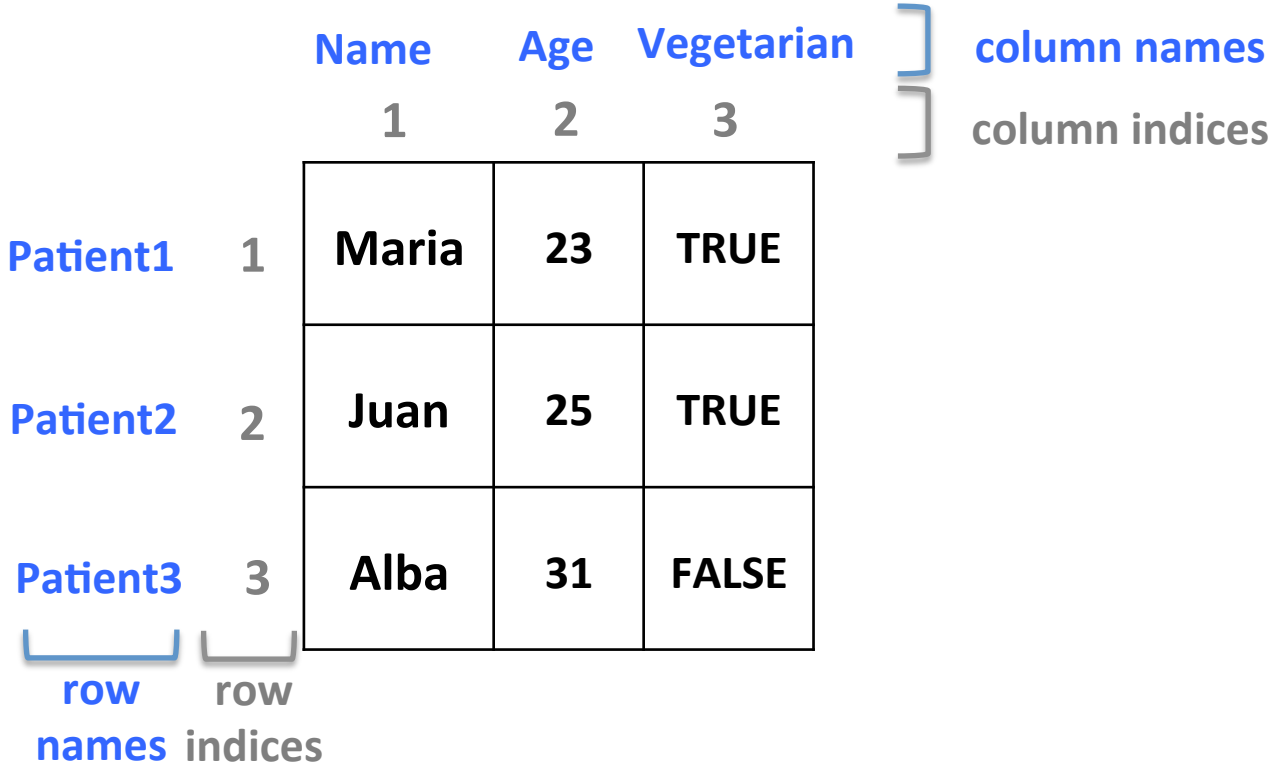
- Column and / or row names can be added to matrices and data frames:

```
colnames(d) <- c("Name", "Age", "Vegetarian")
rownames(d) <- c("Patient1", "Patient2", "Patient3")
```

Dimension names

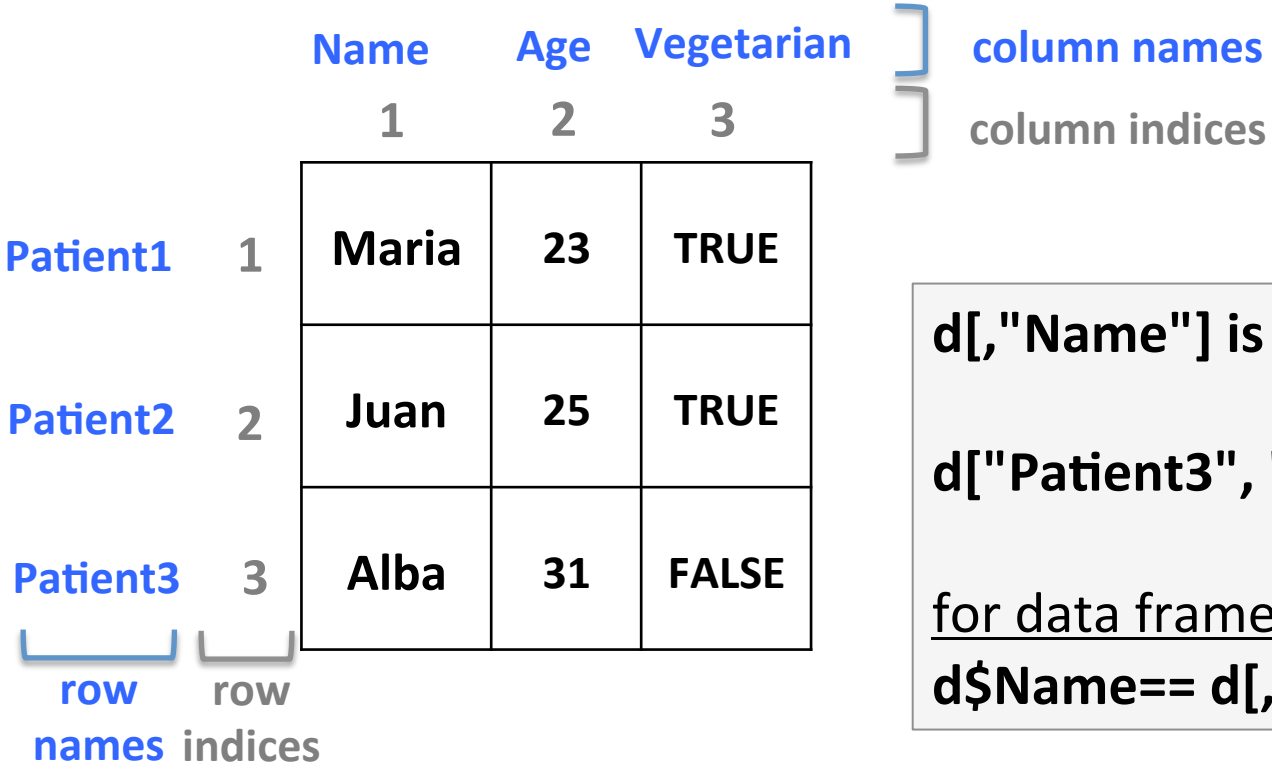
- Instead of indices, you can use dimension names to subset a matrix or a data frame:

d



Dimension names

- Instead of indices, you can use dimension names to subset a matrix or a data frame:



d[,"Name"] is d[,1]

d["Patient3", "Age"] is d[3,2]

for data frames:

d$Name== d[,1] is d[, "Name"]

Create matrices and data frames with names

- **Matrix**

```
m <- matrix(1:4, ncol=2,
    dimnames=list(c("row1", "row2"), c("col1", "col2")))
```

- **Data frame**

```
df <- data.frame(col1=1:2, col2=1:2,
    row.names=c("row1", "row2"))
```

Manipulate 2-dimensional objects

- Same principle as for vectors… but 2-dimensional.

- Select columns of b if elements in **row 3 are inferior or equal to 4:**

b

1	**44**
0	**12**
34	**4**

b[, b[3 ,] <= 4]

44
12
4

Manipulate 2-dimensional objects

- Select rows of b if elements in **column 2 are superior to 24:**

		Name	Age	Vegetarian
		1	2	3
Patient1	1	Maria	23	TRUE
Patient2	2	Juan	25	TRUE
Patient3	3	Alba	31	FALSE

d[d[,2] > 24,]

or

d[d[,"Age"] > 24,]

or

d[d$Age > 24,]

Juan	25	TRUE
Alba	31	FALSE

Manipulate 2-dimensional objects

- Select rows of d if elements in **column 3 are TRUE:**

		Name	Age	Vegetarian
		1	2	3
Patient1	1	Maria	23	TRUE
Patient2	2	Juan	25	TRUE
Patient3	3	Alba	31	FALSE

d[d[, 3] == TRUE,]

Maria	23	TRUE
Juan	25	TRUE

Manipulate 2-dimensional objects

- Select rows of d based on 2 columns:

d[d$Age >= 25 & d$Vegetarian == TRUE,]

		Name	Age	Vegetarian
		1	2	3
Patient1	1	Maria	23	TRUE
Patient2	2	Juan	25	TRUE
Patient3	3	Alba	31	FALSE

Juan	25	TRUE

Some useful commands for matrices and data frames

- **cbind**: adds a column
  cbind(d, 1:3)

- **rbind**: adds a row
  rbind(d, 4:6)

- **rowSums/colSums**: processes the sum of rows/columns of a matrix
  rowSums(b); colSums(b)

Some useful commands for matrices and data frames

- **cbind**: adds a column
  cbind(d, 1:3)

- **rbind**: adds a row
  rbind(d, 4:6)

- **rowSums/colSums**: processes the sum of rows/columns of a matrix
  rowSums(b); colSums(b)

Some useful commands

- Add an entry/patient **to d:**

d <- rbind(d, c("Jordi", 33, FALSE))

Maria	**23**	**TRUE**
Juan	**25**	**TRUE**
Alba	**31**	**FALSE**
Jordi	**33**	**FALSE**

the **apply** function

- Powerful tool to **apply a command** to all rows or all columns of a data frame or a matrix

d

Maria	**23**	**TRUE**
Juan	**25**	**TRUE**
Alba	**31**	**FALSE**

apply(d, 2, table)

$Name
Alba Juan Maria
1 1 1

$Age
23 25 31
1 1 1

$Vegetarian
TRUE FALSE
2 1

Exercise 4:
matrix manipulation
Exercise 5:
Data frame manipulation