# Dataframes: combining different types of values

- A data frame is a generalized matrix, where different columns can have different modes (numeric, character, factor, etc.).

- For example vectors and/or factors of the same length that are related "across", such that data in the same position come from the same experimental unit (subject, animal, etc).

# Dataframes

The function data.frame() allows to create one from scratch

```
> S<-as.factor(c("F","M","M","F"))

> Patients <- data.frame(age=c(31,32,40,50),sex=S)

> Patients
  age sex
1  31   F
2  32   M
3  40   M
4 50   F
```

# Creating a Dataframe from a matrix

- To create a data frame from a matrix use the function as.data.frame()

```
> m<-matrix(1:12, ncol=4, byrow=TRUE)

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

> m.df<-as.data.frame(m)
➢ m.df<-as.data.frame(t(m))
  V1 V2 V3
1  1  5  9
2  2  6 10
3  3  7 11
4  4  8 12
```

# Creating a Dataframe from vectors

- To create a data frame from vectors use the function data.frame()

```
> employee <- c("John Doe","Peter Gynn","Jolie Hope")
> salary <- c(21000, 23400, 26800)
> startdate <-
as.Date(c("2010-11-1","2008-3-25","2007-3-14"))

> employ.data <- data.frame(employee, salary, startdate)
> str(employ.data)
'data.frame':	3 obs. of  3 variables:
 $ employee : Factor w/ 3 levels "John Doe","Jolie Hope",..:
1 3 2
 $ salary   : num  21000 23400 26800
 $ startdate: Date, format: "2010-11-01" "2008-03-25"
"2007-03-14"
```

# Dataframe: keeping character as char

- The original vector employee was a character vector,but R converted it in a factor the data frame

```
> str(employ.data)
```

```
> str(employ.data)
data.frame':   3 obs. of  3 variables:
 $ employee : chr  "John Doe" "Peter Gynn" "Jolie Hope"
 $ salary   : num  21000 23400 26800
 $ startdate: Date, format: "2010-11-01" "2008-03-25"
"2007-03-14"
```

```
> employ.data <- data.frame(employee, salary, startdate,
+                           stringsAsFactors=FALSE)
```

# Looking at a Dataframe

- Structure: str()
- Number of variables: ncols() and length()
- Number of observations: nrow()

```
> m.df<-as.data.frame(t(m))
 V1 V2 V3
1  1  5  9
2  2  6 10
3  3  7 11
4  4  8 12
> str(m.df)
'data.frame':  4 obs. of  3 variables:
 $ V1: int  1 2 3 4
 $ V2: int  5 6 7 8
 $ V3: int  9 10 11 12
> ncol(m.df)
[1] 3
> length(m.df)
[1] 3
> nrow(m.df)
[1] 4
```

# Data frames

```
# Get the structure of the data frame.
> str(emp.data)
'data.frame': 5 obs. of 4 variables:
$ emp_id : int 1 2 3 4 5
$ emp_name : chr "Rick" "Dan" "Michelle" "Ryan" ...
$ salary : num 623 515 611 729 843

# Get the statistical summary of the data with  summary()
>summary(emp.data)
emp_id         emp_name               salary
Min.  :1       Length:5               Min.  :515.2
1st Qu.:2      Class :character       1st Qu.:611.0
Median :3      Mode  :character       Median :623.3
Mean  :3                              Mean  :664.4
3rd Qu.:4                             3rd Qu.:729.0 3rd
Max.  :5                              Max.  :843.2
```

# Indexing a data frame

- A data frame is a generalized matrix and work as such for data indexing

```
> S<-as.factor(c("F","M","M","F"))

> Patients <- data.frame(age=c(31,32,40,50),sex=S)

> Patients
  age sex
1  31   F
2  32   M
3  40   M
4   50   F

> Patients[1,]
  Age gender
1 31       F

> Patients[2,]
  Age gender
2  32       M
```

# Accessing a data frame

- When looking at the result of str() we see that variables are preceded by a $ sign

```
> str(Patients)
'data.frame':    4 obs. of  2 variables:
 $ age: num  31 32 40 50
 $ sex: Factor w/ 2 levels "F","M": 1 2 2 1

> Patients$age
[1] 31 32 40 50

> Patients$sex
[1] F M M F
Levels: F M
```

# Adding rows

```
# Add a new row
> rbind(Patients,c(60,"F"))
  age sex
1  31   F
2  32   M
3  40   M
4  50   F
5  60   F
```

Remember: The two data frames must have the same variables. If dataframe1 has variables that dataframe2 does not have, do one of the following things before joining:

. Delete the extra variables in dataframe1

. Create the additional variables in dataframe2 with value NA (missing)

# Adding columns: merge()

```
d1
  id sex tc
  1 Nam 4.0
  2   Nu 3.5
  3   Nu 4.7
  4 Nam 7.7
  5 Nam 5.0
  6   Nu 4.2
  7 Nam 5.9
  8 Nam 6.1
  9 Nam 5.9
10   Nu 4.0
```

```
d2
  id sex tg
  1 Nam 1.1
  2   Nu 2.1
  3   Nu 0.8
  4 Nam 1.1
  5 Nam 2.1
  6   Nu 1.5
  7 Nam 2.6
  8 Nam 1.5
  9 Nam 5.4
10   Nu 1.9
11   Nu 1.7
```

```
d <- merge(d1, d2, by="id", all=TRUE)
d
   id sex.x  tc sex.y  tg
1   1    Nam 4.0   Nam 1.1
2   2     Nu 3.5    Nu 2.1
3   3     Nu 4.7    Nu 0.8
4   4    Nam 7.7   Nam 1.1
5   5    Nam 5.0   Nam 2.1
6   6     Nu 4.2    Nu 1.5
7   7    Nam 5.9   Nam 2.6
8   8    Nam 6.1   Nam 1.5
9   9    Nam 5.9   Nam 5.4
10 10     Nu 4.0    Nu 1.9
11 11   <NA>  NA    Nu 1.7
```

In most cases, two data frames are joined by one or more common key variables, (e.g. “id”)