# Data Frames

# Data Frames in R

- **Matrizen** sind intern Vektoren, **Data Frames** sind intern Listen
- Wie Listen, können Data Frames mehrere verschiedene Datentypen speichern (sogar Listen oder andere Data Frames)
- Wie Matrizen, haben Data Frames zwei Dimensionen (Zeilen und Spalten)
- Zum Erstellen von Data Frames gibt es den Befehl `data.frame`
- Funktioniert wie erstellen von Listen, aber alle Komponenten/Elemente müssen die selbe Länge haben

# Data Frames in R

```r
students <- data.frame(
  Vorname = c("Alex", "Matthias", "Maria", "Lukas"),
  Nachname = c("Bauer", "Vogel", "Magdalena", "NA"),
  Semester = c(5L, 3L, 9L, 6L)
)

students
##    Vorname  Nachname Semester
## 1     Alex     Bauer        5
## 2 Matthias     Vogel        3
## 3    Maria Magdalena        9
## 4    Lukas        NA        6

str(students)
## 'data.frame':    4 obs. of  3 variables:
##  $ Vorname : chr  "Alex" "Matthias" "Maria" "Lukas"
##  $ Nachname: chr  "Bauer" "Vogel" "Magdalena" "NA"
##  $ Semester: int  5 3 9 6
```

# Indizierung

- Indizierung bei Data Frames ist eine Mischung aus Listen- und Matrixindizierung
- Spalten sind wie Listenkomponenten

```
students[[3]] # 3te Spalte (als Vektor)
## [1] 5 3 9 6

students$Semester # 3te Spalte (als Vektor)
## [1] 5 3 9 6

students[, "Semester"]
## [1] 5 3 9 6

students[3] # 3te Spalte als Data Frame
##   Semester
## 1        5
## 2        3
## 3        9
## 4        6

students[1, 3] # erste Zeile, 3te Spalte (wie bei Matrizen)
## [1] 5
```

# Indizierung

```r
students[1, ] # bleibt DF
##   Vorname Nachname Semester
## 1    Alex    Bauer        5

students[, 3] # wird Vektor
## [1] 5 3 9 6

students[, 3, drop = FALSE] # bleibt DF
##   Semester
## 1        5
## 2        3
## 3        9
## 4        6
```

# Interation

Bei data frames kann man

- mit `lapply`, `sapply`, `vapply` wie bei Listen über Komponenten iterieren (hier Spalten)
- mit `apply` über Zeilen/Spalten iterieren

```
lapply(students, typeof) # always returns list
## $Vorname
## [1] "character"
##
## $Nachname
## [1] "character"
##
## $Semester
## [1] "integer"

sapply(students, typeof) # returns list, vector or matrix, depending on output
##     Vorname    Nachname    Semester
## "character" "character"   "integer"

vapply(students, typeof, character(1)) # returns vector of specified type
##     Vorname    Nachname    Semester
## "character" "character"   "integer"
```

# Interation

Bei data frames kann man

- mit `lapply`, `sapply`, `vapply` wie bei Listen über Komponenten iterieren (hier Spalten)
- mit `apply` über Zeilen/Spalten iterieren

```
apply(students, 2, typeof) # apply always returns matrix or vector
##     Vorname    Nachname    Semester
## "character" "character" "character"

apply(students, 1, function(z) z[1:2])
##          [,1]    [,2]       [,3]        [,4]
## Vorname  "Alex"  "Matthias" "Maria"     "Lukas"
## Nachname "Bauer" "Vogel"    "Magdalena" "NA"
```

# Hinzufügen, Ändern, Löschen

```
# Zeile hinzufügen
students <- rbind(students, data.frame(Vorname = "Merle", Nachname = "Maier", Semester = 4))
students
##    Vorname  Nachname Semester
## 1     Alex     Bauer        5
## 2 Matthias     Vogel        3
## 3    Maria Magdalena        9
## 4    Lukas        NA        6
## 5    Merle     Maier        4

# Spalten hinzufügen
students[["Matrikelnummer"]] <- c("185369", "183959", "183373", "1805317") # zu kurz
## Error in `[[<-.data.frame`(`*tmp*`, "Matrikelnummer", value = c("185369", : replacement has 4 ro
students[["Matrikelnummer"]] <- c("185369", "183959", "183373", "1805317", "180444")
str(students)
## 'data.frame':    5 obs. of  4 variables:
##  $ Vorname      : chr  "Alex" "Matthias" "Maria" "Lukas" ...
##  $ Nachname     : chr  "Bauer" "Vogel" "Magdalena" "NA" ...
##  $ Semester     : num  5 3 9 6 4
##  $ Matrikelnummer: chr  "185369" "183959" "183373" "1805317" ...
```

# Hinzufügen, Ändern, Löschen

```r
# Zeile löschen (wie bei Matrix)
students <- students[-4, ] # Zeile löschen; wie bei Matrix
nrow(students)
## [1] 4

# Spalte löschen (wie bei Liste); alternativ: students <- students[, -4]
students[["Matrikelnummer"]] <- NULL
str(students)
## 'data.frame':    4 obs. of  3 variables:
##  $ Vorname : chr  "Alex" "Matthias" "Maria" "Merle"
##  $ Nachname: chr  "Bauer" "Vogel" "Magdalena" "Maier"
##  $ Semester: num  5 3 9 4
```

# Umsortieren

Sortieren nach Zeilen:

```
students[rev(seq_len(nrow(students))), ]
##    Vorname  Nachname Semester
## 5    Merle     Maier        4
## 3    Maria Magdalena        9
## 2 Matthias     Vogel        3
## 1     Alex     Bauer        5

students[order(students$Semester), ]
##    Vorname  Nachname Semester
## 2 Matthias     Vogel        3
## 5    Merle     Maier        4
## 1     Alex     Bauer        5
## 3    Maria Magdalena        9
```

# Umsortieren

Sortieren nach Spalten:

```
students[, c("Semester", setdiff(names(students), "Semester"))]
##   Semester  Vorname  Nachname
## 1        5     Alex     Bauer
## 2        3 Matthias     Vogel
## 3        9    Maria Magdalena
## 5        4    Merle     Maier

students[, rev(names(students))]
##   Semester  Nachname  Vorname
## 1        5     Bauer     Alex
## 2        3     Vogel Matthias
## 3        9 Magdalena    Maria
## 5        4     Maier    Merle
```

# Helpful functions

```
length(students) # wie bei Liste, drei Komponenten (Spalten)
## [1] 3
nrow(students) # wie bei Matrizen
## [1] 4
ncol(students) # wie bei Matrizen
## [1] 3
dim(students) # wie bei Matrizen
## [1] 4 3
head(students, 2)
##    Vorname Nachname Semester
## 1     Alex    Bauer        5
## 2 Matthias    Vogel        3
tail(students, 2)
##   Vorname  Nachname Semester
## 3   Maria Magdalena        9
## 5   Merle     Maier        4
summary(students)
##    Vorname            Nachname           Semester
##  Length:4           Length:4           Min.   :3.00
##  Class :character   Class :character   1st Qu.:3.75
##  Mode  :character   Mode  :character   Median :4.50
##                                        Mean   :5.25
##                                        3rd Qu.:6.00
##                                        Max.   :9.00
```