# Part 1: R Foundations

(d) Data Frames and Tibbles

# Recap - R Data Types

| | Homogenous | Heterogenous |
|---|---|---|
| 1d | Atomic Vector | List |
| 2d | Matrix | **Data Frame/Tibble** |
| nd | Array | |

- The most common way of storing data in R
- Under the hood, a data frame is a list of equal-length vectors
- A two-dimensional structure, it shares properties of both a list and a matrix

# Creating a data frame

```
d <- data.frame(x=1:3, y = LETTERS[1:3], z = letters[1:3])
d
```

```
##   x y z
## 1 1 A a
## 2 2 B b
## 3 3 C c
```

```
d$x
```

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

```
d$y
```

```
## [1] A B C
## Levels: A B C
```

```
d$z
```

```
## [1] a b c
## Levels: a b c
```

# summary function with data frames

```
d <- data.frame(x=1:3, y = LETTERS[1:3],
                z = letters[1:3])
d
```

```
##   x y z
## 1 1 A a
## 2 2 B b
## 3 3 C c
```

```
summary(d)
```

```
##        x       y     z
##  Min.   :1.0   A:1   a:1
##  1st Qu.:1.5   B:1   b:1
##  Median :2.0   C:1   c:1
##  Mean   :2.0
##  3rd Qu.:2.5
##  Max.   :3.0
```

# mtcars data frame

A data frame with 32 observations on 11 variables.

- **mpg** Miles/(US) gallon
- **cyl** Number of cylinders
- **disp** Displacement (cu.in.)
- **hp** Gross horsepower
- **drat** Rear axle ratio
- **wt** Weight (1000 lbs)
- **qsec** 1/4 mile time
- **vs** V/S
- **am** Transmission (0 = automatic, 1 = manual)
- **gear** Number of forward gears
- **carb** Number of carburetors

# mtcars sample data

```
knitr::kable(mtcars[1:10,1:6])
```

| | mpg | cyl | disp | hp | drat | wt |
|---|---|---|---|---|---|---|
| Mazda RX4 | 21.0 | 6 | 160.0 | 110 | 3.90 | 2.620 |
| Mazda RX4 Wag | 21.0 | 6 | 160.0 | 110 | 3.90 | 2.875 |
| Datsun 710 | 22.8 | 4 | 108.0 | 93 | 3.85 | 2.320 |
| Hornet 4 Drive | 21.4 | 6 | 258.0 | 110 | 3.08 | 3.215 |
| Hornet Sportabout | 18.7 | 8 | 360.0 | 175 | 3.15 | 3.440 |
| Valiant | 18.1 | 6 | 225.0 | 105 | 2.76 | 3.460 |
| Duster 360 | 14.3 | 8 | 360.0 | 245 | 3.21 | 3.570 |
| Merc 240D | 24.4 | 4 | 146.7 | 62 | 3.69 | 3.190 |
| Merc 230 | 22.8 | 4 | 140.8 | 95 | 3.92 | 3.150 |
| Merc 280 | 19.2 | 6 | 167.6 | 123 | 3.92 | 3.440 |

# mtcars using **str()**

```
str(mtcars)
```

```
## 'data.frame':    32 obs. of  11 variables:
##  $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 1
##  $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
##  $ disp: num  160 160 108 258 360 ...
##  $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
##  $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92
##  $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
##  $ qsec: num  16.5 17 18.6 19.4 17 ...
##  $ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
##  $ am  : num  1 1 1 0 0 0 0 0 0 0 ...
##  $ gear: num  4 4 4 3 3 3 3 4 4 4 ...
##  $ carb: num  4 4 1 1 2 1 4 2 2 4 ...
```

# head() and tail() functions

```r
head(mtcars[,1:6])
```

```
##                   mpg cyl disp  hp drat    wt
## Mazda RX4         21.0   6  160 110 3.90 2.620
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875
## Datsun 710        22.8   4  108  93 3.85 2.320
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215
## Hornet Sportabout 18.7   8  360 175 3.15 3.440
## Valiant           18.1   6  225 105 2.76 3.460
```

```r
tail(mtcars[,1:6])
```

```
##                 mpg cyl  disp  hp drat    wt
## Porsche 914-2  26.0   4 120.3  91 4.43 2.140
## Lotus Europa   30.4   4  95.1 113 3.77 1.513
## Ford Pantera L 15.8   8 351.0 264 4.22 3.170
## Ferrari Dino   19.7   6 145.0 175 3.62 2.770
## Maserati Bora  15.0   8 301.0 335 3.54 3.570
## Volvo 142E     21.4   4 121.0 109 4.11 2.780
```

# Subsetting rows

```
mtcars[mtcars$gear == 5,]
```

```
##                  mpg cyl  disp  hp drat    wt qsec vs am
## Porsche 914-2   26.0   4 120.3  91 4.43 2.140 16.7  0  1
## Lotus Europa    30.4   4  95.1 113 3.77 1.513 16.9  1  1
## Ford Pantera L 15.8   8 351.0 264 4.22 3.170 14.5  0  1
## Ferrari Dino    19.7   6 145.0 175 3.62 2.770 15.5  0  1
## Maserati Bora   15.0   8 301.0 335 3.54 3.570 14.6  0  1
```

# Accessing rows/columns

```
mtcars[1:10,1:6]
```

```
##                    mpg cyl  disp  hp drat    wt
## Mazda RX4         21.0   6 160.0 110 3.90 2.620
## Mazda RX4 Wag     21.0   6 160.0 110 3.90 2.875
## Datsun 710        22.8   4 108.0  93 3.85 2.320
## Hornet 4 Drive    21.4   6 258.0 110 3.08 3.215
## Hornet Sportabout 18.7   8 360.0 175 3.15 3.440
## Valiant           18.1   6 225.0 105 2.76 3.460
## Duster 360        14.3   8 360.0 245 3.21 3.570
## Merc 240D         24.4   4 146.7  62 3.69 3.190
## Merc 230          22.8   4 140.8  95 3.92 3.150
## Merc 280          19.2   6 167.6 123 3.92 3.440
```

# Filtering rows and columns

```
mtcars[mtcars$cyl == 6,c("mpg","cyl")]
```

```
##                    mpg cyl
## Mazda RX4         21.0   6
## Mazda RX4 Wag     21.0   6
## Hornet 4 Drive    21.4   6
## Valiant           18.1   6
## Merc 280          19.2   6
## Merc 280C         17.8   6
## Ferrari Dino      19.7   6
```

# Challenge 1.5

- List all the cars that have an **mpg** greater than the average
- List the car(s) with the greatest displacement (**disp**)

# Adding new columns to a data frame

- Often the initial data set may not contain sufficient information for analysis
- Adding new variables (columns) is an important feature to have
- Data frames support this: columns can be combined or new information used

```
mtcars$name <- rownames(mtcars)
mtcars[1:5,-(1:8)]
```

```
##                   am gear carb              name
## Mazda RX4          1    4    4         Mazda RX4
## Mazda RX4 Wag      1    4    4     Mazda RX4 Wag
## Datsun 710         1    4    1        Datsun 710
## Hornet 4 Drive     0    3    1    Hornet 4 Drive
## Hornet Sportabout  0    3    2 Hornet Sportabout
```

# Challenge 1.6

Create a new column on mtcars that contains kilometers per gallon.

# Missing data - complete.cases()

```
d <- data.frame(x=1:3, y = LETTERS[1:3],
                z = letters[1:3])
d[2,3] <- NA
d
```

```
##   x y    z
## 1 1 A    a
## 2 2 B <NA>
## 3 3 C    c
```

```
complete.cases(d)
```

```
## [1]  TRUE FALSE  TRUE
```

```
d[complete.cases(d),]
```

```
##   x y z
## 1 1 A a
## 3 3 C c
```

# The tibble

- Tibbles are data frames, but they tweak some older behaviours to make life a little easier
- One of the unifying features of the tidyverse
- To coerce a data frame to a tibble, use as_tibble()
- A tibble can be created from individual vectors using tibble()

```
t <- tibble(x=1:3, y = LETTERS[1:3], z = letters[1:3])
t
```

```
## # A tibble: 3 x 3
##       x y     z
##   <int> <chr> <chr>
## 1     1 A     a
## 2     2 B     b
## 3     3 C     c
```

# Tibble abbreviations

```
t
```

```
## # A tibble: 3 x 3
##       x y     z
##   <int> <chr> <chr>
## 1     1 A     a
## 2     2 B     b
## 3     3 C     c
```

| Abbreviation | Data Type |
|---|---|
| int | integers |
| dbl | double (numeric) |
| chr | character vectors |
| dttm | date-times |
| fctr | categorical |
| date | dates |

# Summary - Part 1: R Foundations

| | Homogenous | Heterogenous |
|---|---|---|
| 1d | Atomic Vector | List |
| 2d | *Matrix* | Data Frame/Tibble |
| nd | *Array* | |

- Atomic Vectors
- Lists
- Functions and Functionals
- Data Frames & Tibbles

# Objects in R

- "Everything that exists in R is an object". Chambers (2008)
- However, while everything is an object, not everything is object-oriented (Wickham 2019)
- Base objects come from S, and were developed before anyone thought that S might need an OOP system. **typeof()** provides information on the base object, and **sloop::otype()**

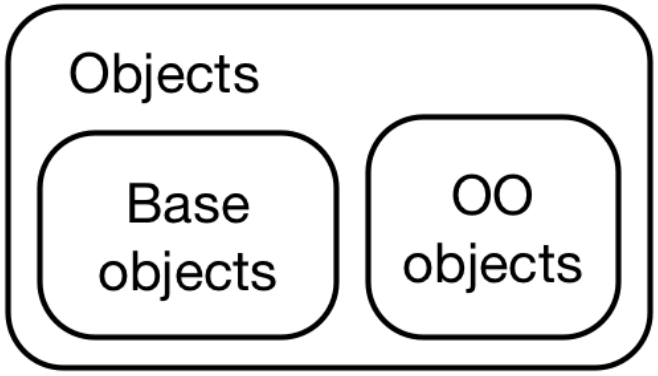


Figure 1: Objects in R

# Examples

```
typeof(1:10)
```

```
## [1] "integer"
```

```
sloop::otype(1:10)
```

```
## [1] "base"
```

```
mod <- lm(eruptions ~ waiting, data=faithful)
typeof(mod)
```

```
## [1] "list"
```

```
sloop::otype(mod)
```

```
## [1] "S3"
```

```
class(mod)
```

```
## [1] "lm"
```

# Test Slide with Plot