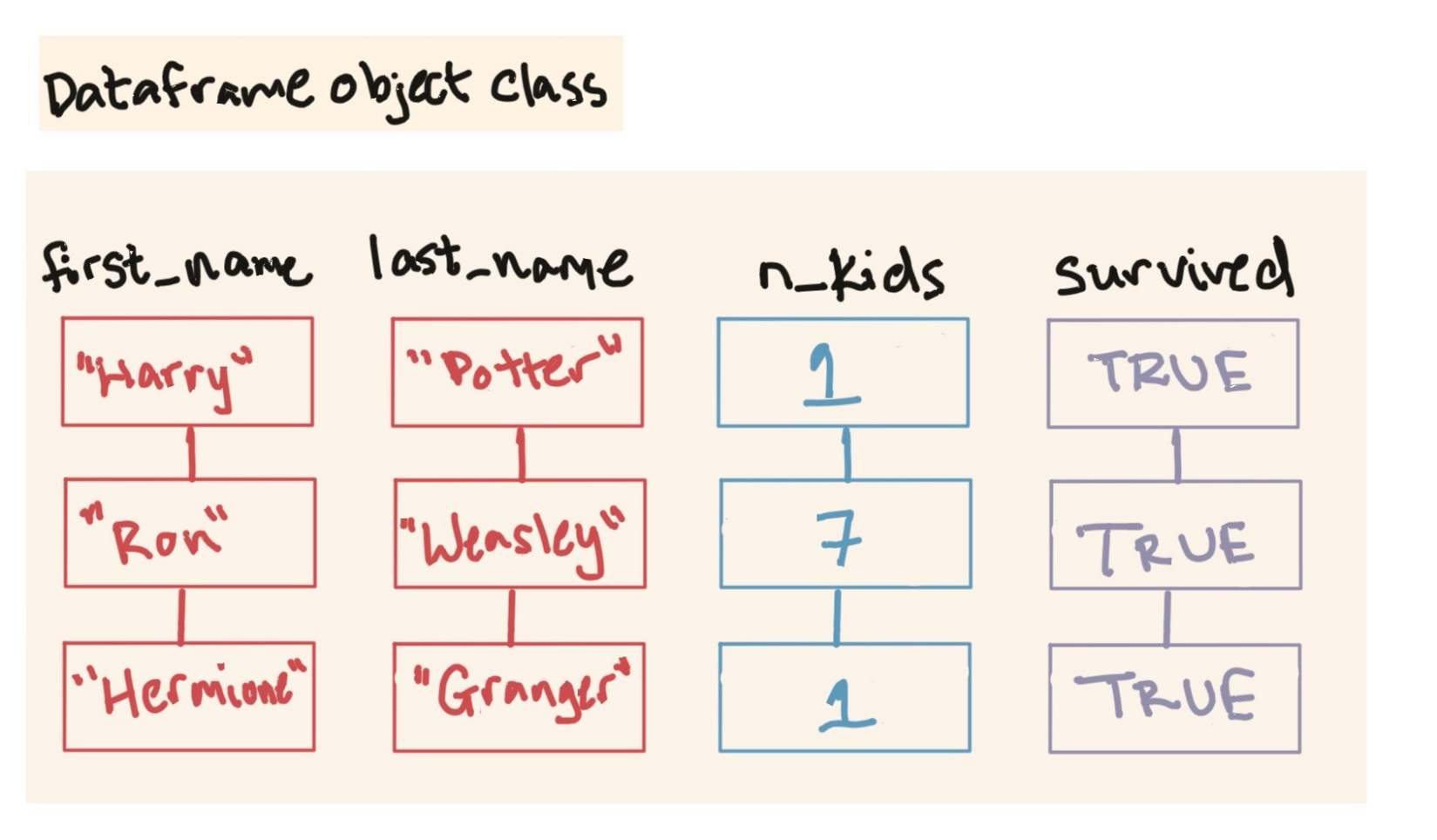
# Preliminaries in R

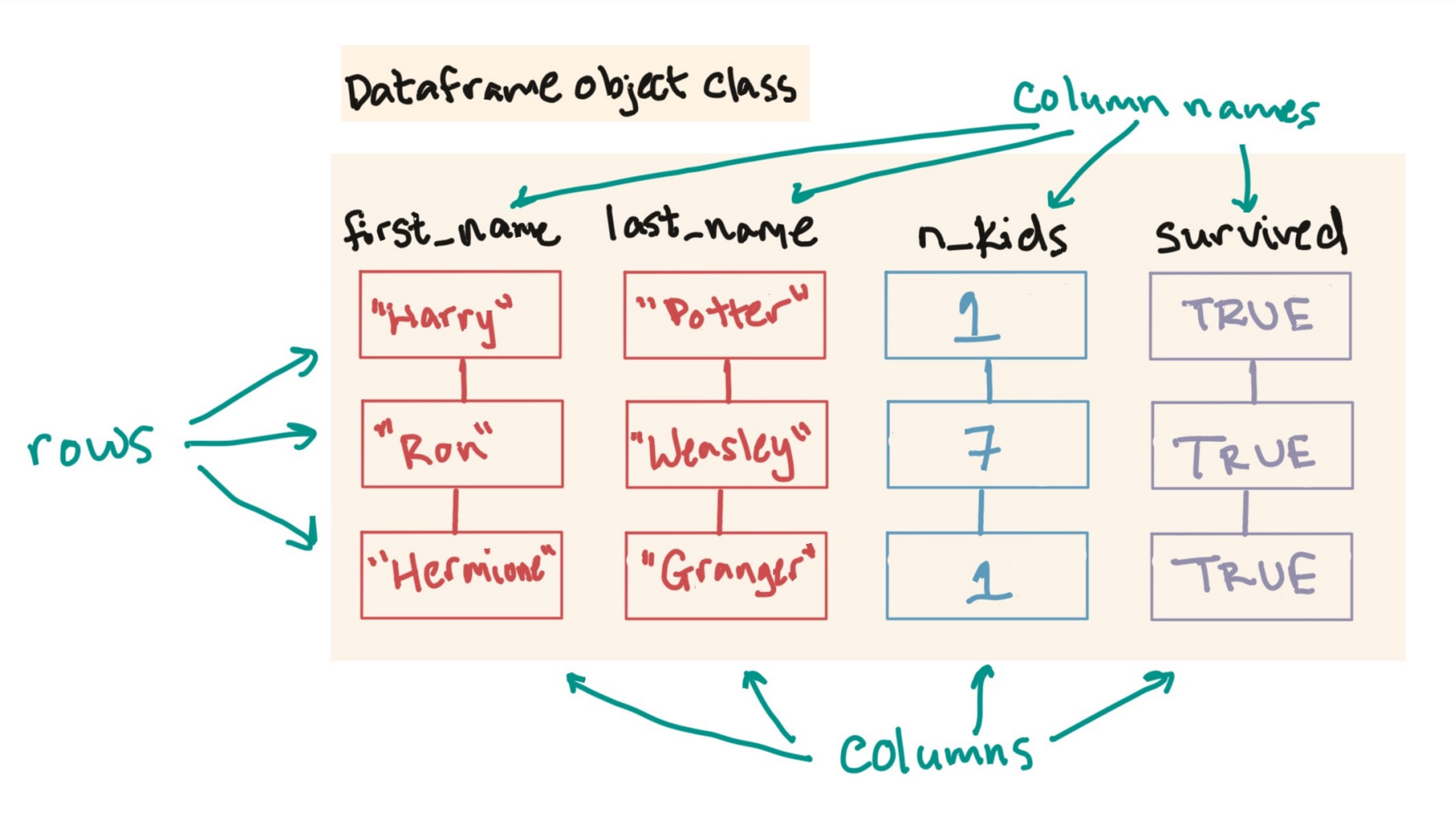
# Dataframes

# Structure of dataframe objects




A **dataframe** combines one or more vectors of the same length stuck together side-by-side.

# Structure of dataframe objects




A dataframe contains **rows** and **columns**, and each column has a **column name**.

# Creating dataframes

We'll be working with a specific class of dataframe called a **tibble**.

You can create tibble dataframe using the `tibble` function from the `tibble` package.

However, most often you will create a dataframe by reading in data from a file—most datasets will not be short enough that you want to enter them by hand in R.

We'll look at both methods of creating dataframes.

# Creating dataframes

The format for creating a tibble dataframe using the `tibble` function is:

```
## Generic code
library(package = "tibble")
[name of object] <- tibble([1st column name] =
                             [1st column content],
                           [2nd column name] =
                             [2nd column content],
                           etc.)
```

# Creating dataframes

```
library(package = "tibble")
hp_data <- tibble(first_name = c("Harry", "Ron", "Hermione"),
                  last_name = c("Potter", "Weasley", "Granger"),
                  n_kids = c(1, 7, 1),
                  survived = c(TRUE, TRUE, TRUE))
hp_data
```

```
## # A tibble: 3 x 4
##   first_name last_name n_kids survived
##   <chr>      <chr>      <dbl> <lgl>
## 1 Harry      Potter         1 TRUE
## 2 Ron        Weasley        7 TRUE
## 3 Hermione   Granger        1 TRUE
```

# Creating dataframes

You can also create dataframes by joining together vector objects you previously created, as long as they have the same length and line up:

```
main_characters <- c("Harry", "Ron", "Hermione")
n_kids <- c(1, 7, 1)
hp_data <- tibble(first_name = main_characters,
                  last_name = c("Potter", "Weasley", "Granger"),
                  n_kids = n_kids,
                  survived = c(TRUE, TRUE, TRUE))
hp_data
```

```
## # A tibble: 3 x 4
##   first_name last_name n_kids survived
##   <chr>      <chr>      <dbl> <lgl>
## 1 Harry      Potter         1 TRUE
## 2 Ron        Weasley        7 TRUE
## 3 Hermione   Granger        1 TRUE
```

# Creating dataframes

Usually, instead of creating a dataframe from vectors, you'll read one in from data on an outside file, for example using `read_csv` from the `readr` package.

For example, to read in a dataset from a csv file called "daily_show_guests.csv":

```
library(package = "readr")
daily_show <- read_csv(file = "daily_show_guests.csv",
                       skip = 4)
```

# Creating dataframes

This has read data in from the external file into a dataframe object in my R session:

```
ls()
```

```
## [1] "daily_show"      "hp_data"         "main_characters" "n_
```

# Dataframes

You can use the functions `dim`, `nrow`, and `ncol` to figure out the dimensions (number of rows and columns) of a dataframe:

```
dim(x = daily_show)
```

```
## [1] 2693    5
```

```
nrow(x = daily_show)
```

```
## [1] 2693
```

```
ncol(x = daily_show)
```

```
## [1] 5
```

# Dataframes

Base R also has some useful functions for quickly exploring dataframes:

- `str`: Show the structure of an R object, including a dataframe
- `summary`: Give summaries of each column of a dataframe.

# Extracting values from dataframes

The `dplyr` package has two functions for extracting data from dataframes by position: `slice` to extract rows based on their row position and select to extract columns based on their column position.

# Extracting values from dataframes

For example, if you wanted to get the first two rows of the `hp_data` dataframe, you could run:

```
library(package = "dplyr")
slice(.data = hp_data, c(1:2))
```

```
## # A tibble: 2 x 4
##   first_name last_name n_kids survived
##   <chr>      <chr>      <dbl> <lgl>
## 1 Harry      Potter         1 TRUE
## 2 Ron        Weasley        7 TRUE
```

# Extracting values from dataframes

If you wanted to get the first and fourth columns, you could run:

```
select(.data = hp_data, c(1, 4))
```

```
## # A tibble: 3 x 2
##   first_name survived
##   <chr>      <lgl>
## 1 Harry      TRUE
## 2 Ron        TRUE
## 3 Hermione   TRUE
```

# Extracting values from dataframes

You can compose calls from both functions. For example, you could extract the values in the first and fourth columns of the first two rows with:

```
select(slice(.data = hp_data, c(1:2)), c(1, 4))
```

```
## # A tibble: 2 x 2
##   first_name survived
##   <chr>      <lgl>
## 1 Harry      TRUE
## 2 Ron        TRUE
```