

# Data I/O + Structure

Data Wrangling in R

## Data Input

- ▶ Sometimes you get weird messages when reading in data:
- ▶ The `spec()` and `problems()` functions show you the specification of how the data was read in.

```
dim(problems(ufo))
```

```
[1] 199    5
```

```
spec(ufo)
```

```
cols(  
  datetime = col_character(),  
  city = col_character(),  
  state = col_character(),  
  country = col_character(),  
  shape = col_character(),  
  `duration (seconds)` = col_double(),  
  `duration (hours/min)` = col_character(),  
  comments = col_character(),  
  `date posted` = col_character().
```

## Data Input: Checking for problems

- ▶ The `stop_for_problems()` function will stop if your data had an error when reading in. If this occurs, you can either use `col_types` (from `spec()`) for the problematic columns, or set `guess_max = Inf` (takes much longer):

```
stop_for_problems(ufo)
```

## More ways to save: write\_rds

If you want to save **one** object, you can use `readr::write_rds` to save to a compressed rds file:

```
write_rds(ufo, path = "ufo_dataset.rds", compress = "xz")
```

## More ways to save: read\_rds

To read this back in to R, you need to use read\_rds, but **need to assign it**:

```
ufo3 = read_rds(path = "ufo_dataset.rds")  
identical(ufo, ufo3) # test if they are the same
```

```
[1] TRUE
```

## More ways to save: save

The save command can save a set of R objects into an “R data file”, with the extension .rda or .RData.

```
x = 5  
save(ufo, x, file = "ufo_data.RData")
```

## More ways to save: load

The opposite of `save` is `load`. The `ls()` command lists the items in the workspace/environment and `rm` removes them:

## What did I just read in?

- ▶ `nrow()` displays the number of rows of a data frame
- ▶ `ncol()` displays the number of columns
- ▶ `dim()` displays a vector of length 2: # rows, # columns

```
dim(ufo)
```

```
[1] 88875    11
```

```
nrow(ufo)
```

```
[1] 88875
```

```
ncol(ufo)
```

```
[1] 11
```



# Data Summaries

- ▶ `colnames()` displays the column names (if any) and `rownames()` displays the row names (if any)
- ▶ Note that tibbles do not have row names

```
colnames(ufo)
```

[1]	"datetime"	"city"	"state"
[4]	"country"	"shape"	"duration"
[7]	"duration (hours/min)"	"comments"	"date posted"
[10]	"latitude"	"longitude"	

# Data Output

While its nice to be able to read in a variety of data formats, it's equally important to be able to output data somewhere.

`write_delim()`: Write a data frame to a delimited file “This is about twice as fast as `write.csv()`, and never writes row names.”

```
args(readr::write_delim)
```

```
function (x, path, delim = " ", na = "NA", append = FALSE,  
          quote_escape = "double")  
NULL
```

# Data Output

`x`: A data frame to write to disk

`path`: the file name where you want to R object written. It can be an absolute path, or a filename (which writes the file to your working directory)

`delim`: what character separates the columns?

- ▶ `","` = .csv - Note there is also `write_csv()` function
- ▶ `"\t"` = tab delimited

## Data Output

For example, we can write back out the Monuments dataset with the new column name:

```
write_csv(ufo[1:100,], path="ufo_first100.csv")
```