The structure of data

Research Methods and Skills

02/11/2021

Writing R Scripts

Scripts are text documents that contain a sequence of commands to be executed sequentially.

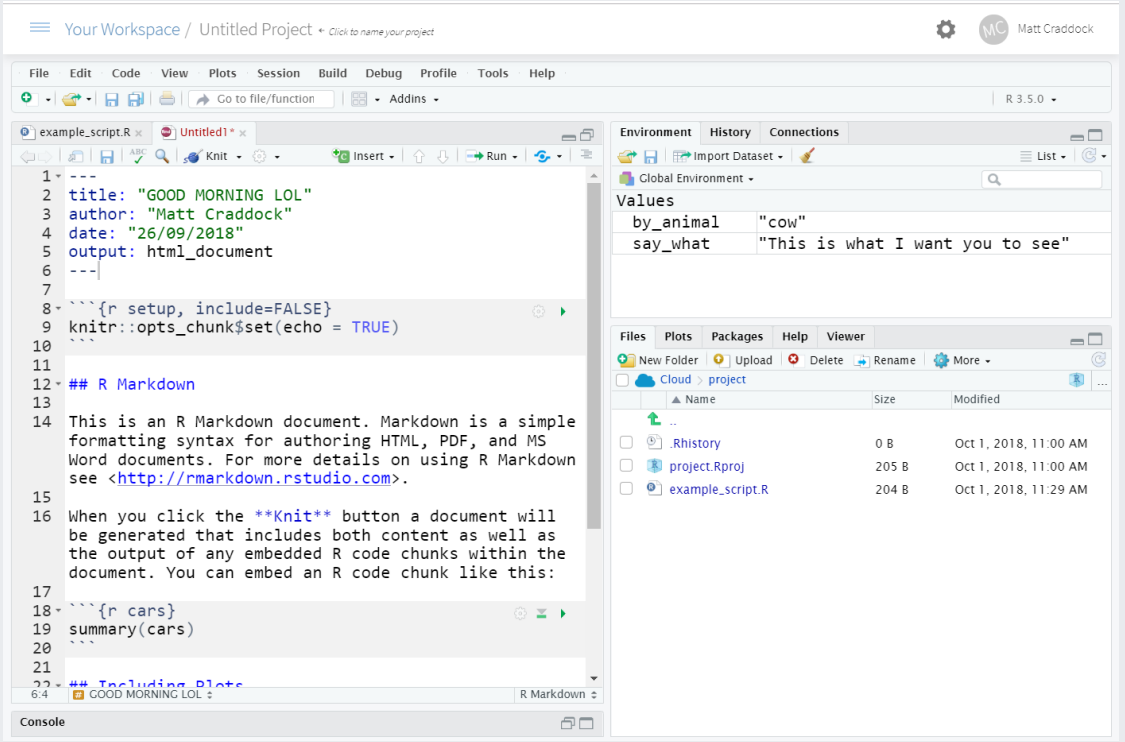
A typical script looks something like this:

```
# Load in required packages using library()
library(tidyverse)

# Define any custom functions here (we haven't covered this!)

# Now load any data you want to work on. (again, we'll cover this later!)
test_data <-
  read_csv("data/a-random-RT-file.csv") %>% # I'll explain what %>% means later
  rename(RT = `reaction times`)

# The rest of the script then runs whatever analyses or plotting you want to do
ggplot(test_data,
       aes(x = RT,
           fill = viewpoint)) +
  geom_density()
```

RMarkdown

RMarkdown documents contain a mixture of code and plain text.

They can be used to produce *reports* and fully formatted documents with whatever structure you choose.



Basic data types

There are five basic data types in R:

Type	Description	Examples
integer	Whole numbers	1, 2, 3
numeric	Any real number, fractions	3.4, 2, -2.3
character	Text	"Hi there", "8.5", "ABC123"
logical	Assertion of truth/falsity	TRUE, FALSE
complex	Real and imaginary numbers	0.34+5.3i

Containers

Vectors are one-dimensional collections of values of the same basic data type.

Matrices are two-dimensional collections of values of the same basic data type.

Lists are collections of objects of varying length and type.

Data frames are tables of data.

Accessing elements from containers

You can use the **[]** operator after the name of an object to extract indvidual elements from that object.

```
one_to_four
```

```
##    Monday   Tuesday Wednesday  Thursday
##         1         2         3         4
```

```
one_to_four["Wednesday"]
```

```
## Wednesday
##         3
```

```
test_matrix
```

```
##             [,1]       [,2]       [,3]
## [1,] 0.2353145  1.3929560  0.2605482
## [2,] 1.0336660 -0.8899093 -0.9559970
## [3,] 0.4376461  1.3994929  0.7738423
```

```
test_matrix[2:3, 1:2]
```

```
##             [,1]       [,2]
## [1,] 1.0336660 -0.8899093
## [2,] 0.4376461  1.3994929
```


tidyverse
www.rstudio.com

Tidyverse

The **tidyverse** is a collection of packages that expand R's functions in a structured, coherent way.

```
install.packages("tidyverse")
```

There are eight core **tidyverse** packages loaded using **library(tidyverse)**.

- ggplot2
- **tidyr**
- dplyr
- **tibble**
- purrr
- readr
- stringr
- forcats

Tidyverse

You can load all these packages at once.

```
library(tidyverse) # This loads all the tidyverse packages at once
```

You can also load each one individually. We'll be using the **tibble** package next.

```
library(tibble)
```

Many of the *tidyverse* packages create or output *tibbles*, which are essentially a more user-friendly version of data frames.

Tibbles

You can create a *tibble* similar to how you create a data frame, using **tibble()**.

```
age_tibb <-
  tibble(Participant = 1:10,
         cond1 = rnorm(10),
         age_group = rep(c("Old", "Young"),
                         each = 5))
head(age_tibb)
```

```
## # A tibble: 6 x 3
##   Participant  cond1 age_group
##         <int>  <dbl> <chr>
## 1           1  1.39  Old
## 2           2 -0.212 Old
## 3           3 -0.535 Old
## 4           4  0.653 Old
## 5           5  0.459 Old
## 6           6  1.30  Young
```

Tibbles

```
age_tibb <-
  tibble(Participant = 1:10,
         cond1 = rnorm(10),
         age_group = rep(c("Old", "Young"), each = 5))
```

Here I used the **rep()** function to generate a character vector with the values "Old" and "Young".

```
rep(c("Old", "Young"), each = 5)
```

```
##  [1] "Old"   "Old"   "Old"   "Old"   "Old"   "Young" "Young" "Young" "Young"
## [10] "Young"
```

```
rep(c("Old", "Young"), 5)
```

```
##  [1] "Old"   "Young" "Old"   "Young" "Old"   "Young" "Old"   "Young" "Old"
## [10] "Young"
```

Importing data into R

Different types of file

Data comes in many different shapes, sizes, and formats.

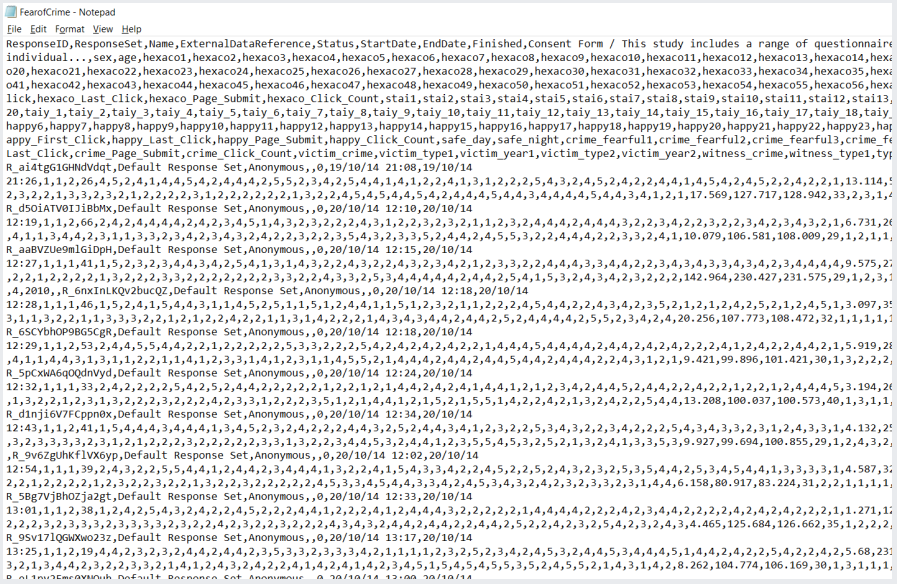
The most common file formats you'll deal with are either Excel files or text files, but you may also find dealing with SPSS files useful.

Fortunately, R has several functions and packages for importing data!

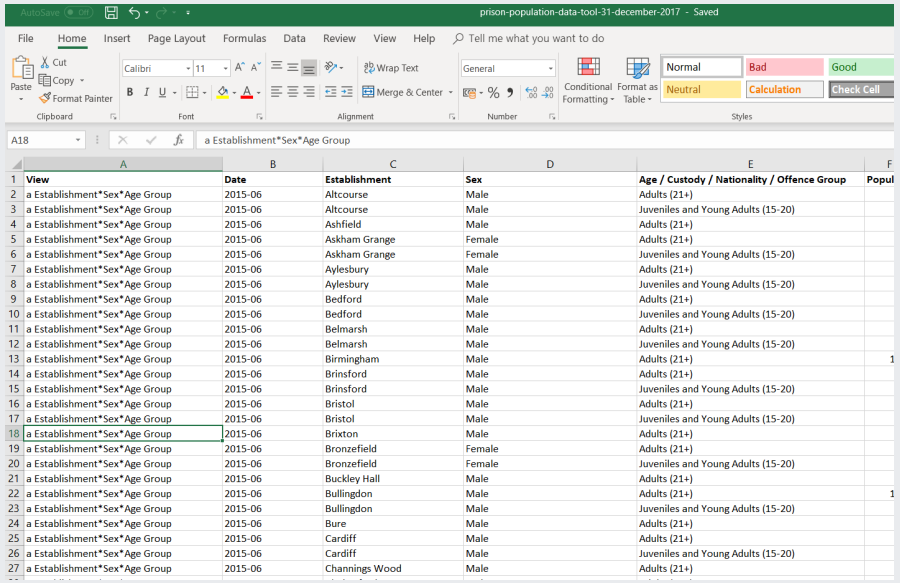
File formats	File extension	Functions	Package
SPSS	.sav	**read_sav()**	library(haven)
Excel	.xls, .xlsx	**read_excel()**	library(readxl)
Text	.csv, .txt, .*	**read_csv()**, **read_delim()**	library(readr)

Importing data into R

Comma-separated values



Excel spreadsheets



	A	B	C	D	E	F
1	**View**	**Date**	**Establishment**	**Sex**	**Age / Custody / Nationality / Offence Group**	**Popul**
2	a Establishment*Sex*Age Group	2015-06	Altcourse	Male	Adults (21+)	
3	a Establishment*Sex*Age Group	2015-06	Altcourse	Male	Juveniles and Young Adults (15-20)	
4	a Establishment*Sex*Age Group	2015-06	Ashfield	Male	Adults (21+)	
5	a Establishment*Sex*Age Group	2015-06	Askham Grange	Female	Adults (21+)	
6	a Establishment*Sex*Age Group	2015-06	Askham Grange	Female	Juveniles and Young Adults (15-20)	
7	a Establishment*Sex*Age Group	2015-06	Aylesbury	Male	Adults (21+)	
8	a Establishment*Sex*Age Group	2015-06	Aylesbury	Male	Juveniles and Young Adults (15-20)	
9	a Establishment*Sex*Age Group	2015-06	Bedford	Male	Adults (21+)	
10	a Establishment*Sex*Age Group	2015-06	Bedford	Male	Juveniles and Young Adults (15-20)	
11	a Establishment*Sex*Age Group	2015-06	Belmarsh	Male	Adults (21+)	
12	a Establishment*Sex*Age Group	2015-06	Belmarsh	Male	Juveniles and Young Adults (15-20)	
13	a Establishment*Sex*Age Group	2015-06	Birmingham	Male	Adults (21+)	1
14	a Establishment*Sex*Age Group	2015-06	Brinsford	Male	Adults (21+)	
15	a Establishment*Sex*Age Group	2015-06	Brinsford	Male	Juveniles and Young Adults (15-20)	
16	a Establishment*Sex*Age Group	2015-06	Bristol	Male	Adults (21+)	
17	a Establishment*Sex*Age Group	2015-06	Bristol	Male	Juveniles and Young Adults (15-20)	
18	a Establishment*Sex*Age Group	2015-06	Brixton	Male	Adults (21+)	
19	a Establishment*Sex*Age Group	2015-06	Bronzefield	Female	Adults (21+)	
20	a Establishment*Sex*Age Group	2015-06	Bronzefield	Female	Juveniles and Young Adults (15-20)	
21	a Establishment*Sex*Age Group	2015-06	Buckley Hall	Male	Adults (21+)	
22	a Establishment*Sex*Age Group	2015-06	Bullingdon	Male	Adults (21+)	1
23	a Establishment*Sex*Age Group	2015-06	Bullingdon	Male	Juveniles and Young Adults (15-20)	
24	a Establishment*Sex*Age Group	2015-06	Bure	Male	Adults (21+)	
25	a Establishment*Sex*Age Group	2015-06	Cardiff	Male	Adults (21+)	
26	a Establishment*Sex*Age Group	2015-06	Cardiff	Male	Juveniles and Young Adults (15-20)	
27	a Establishment*Sex*Age Group	2015-06	Channings Wood	Male	Adults (21+)	

Fear of Crime Dataset

Ellis & Renouf (2018) - the relationship between fear of crime and various personality measures.

Their data is openly available, stored as text in a *comma-separated-values* format (*.csv*).

Once again, we can use the import button or some code (with **read_csv()**) to load this data in and automatically format it into a *tibble*.

```
library(readr)
FearofCrime <- read_csv("data/FearofCrime.CSV")
```

See also Ellis & Merdian, 2015, Frontiers in Psychology

Fear of Crime Dataset

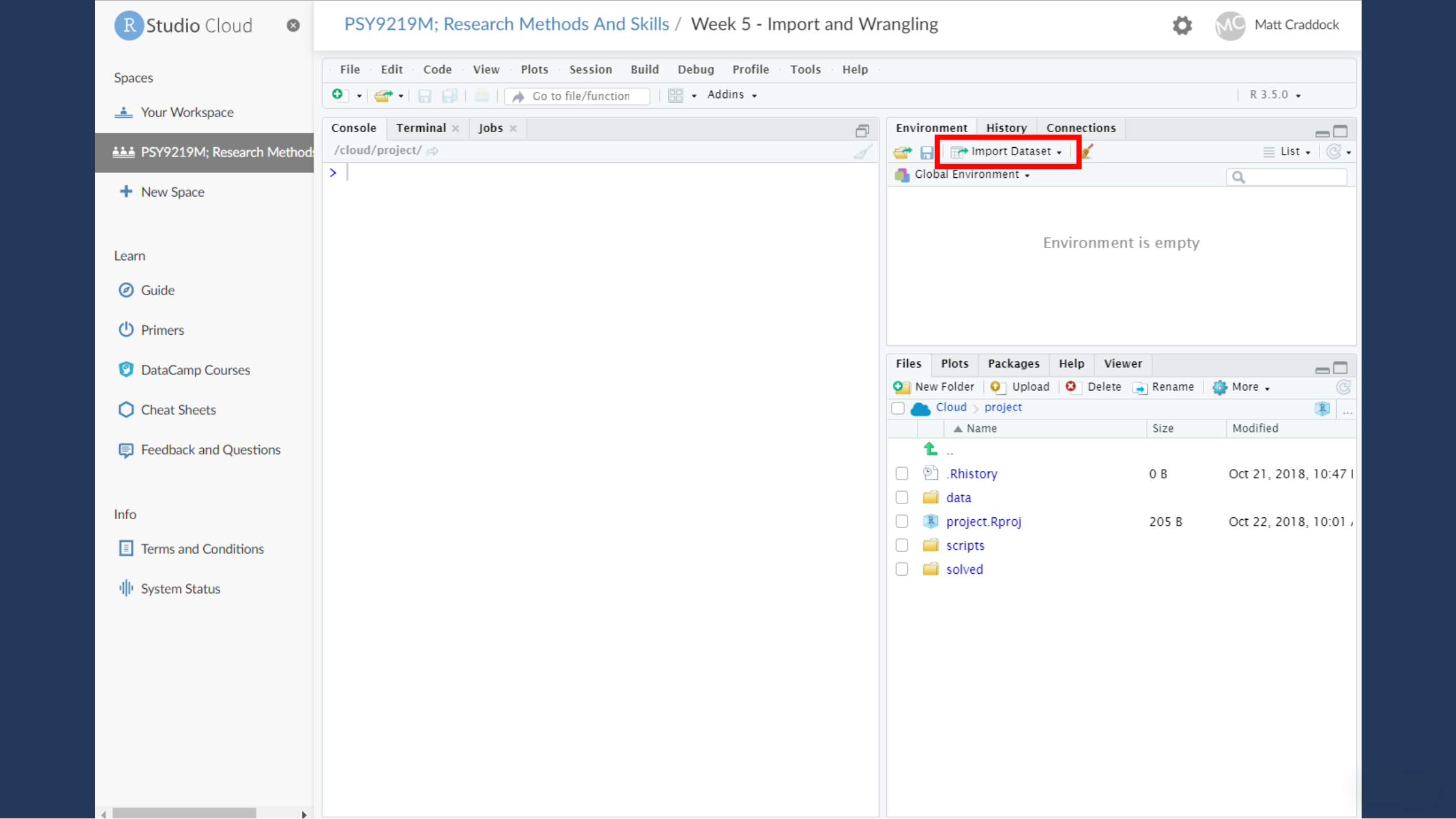
Ellis & Renouf (2018) collected data online using Qualtrics.

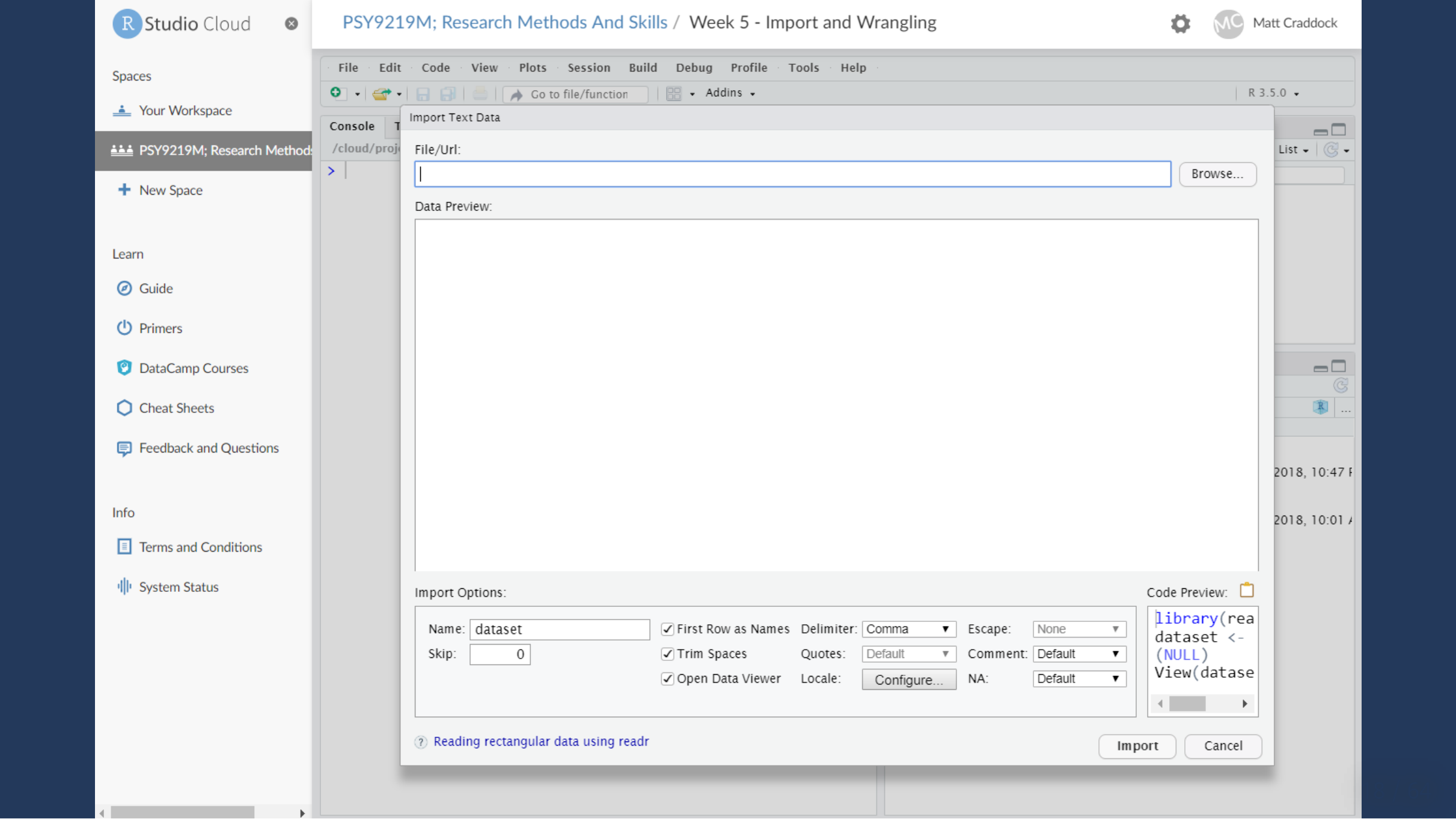
The file contains one column for each question that the participants answered, for a total of 169(!) columns.

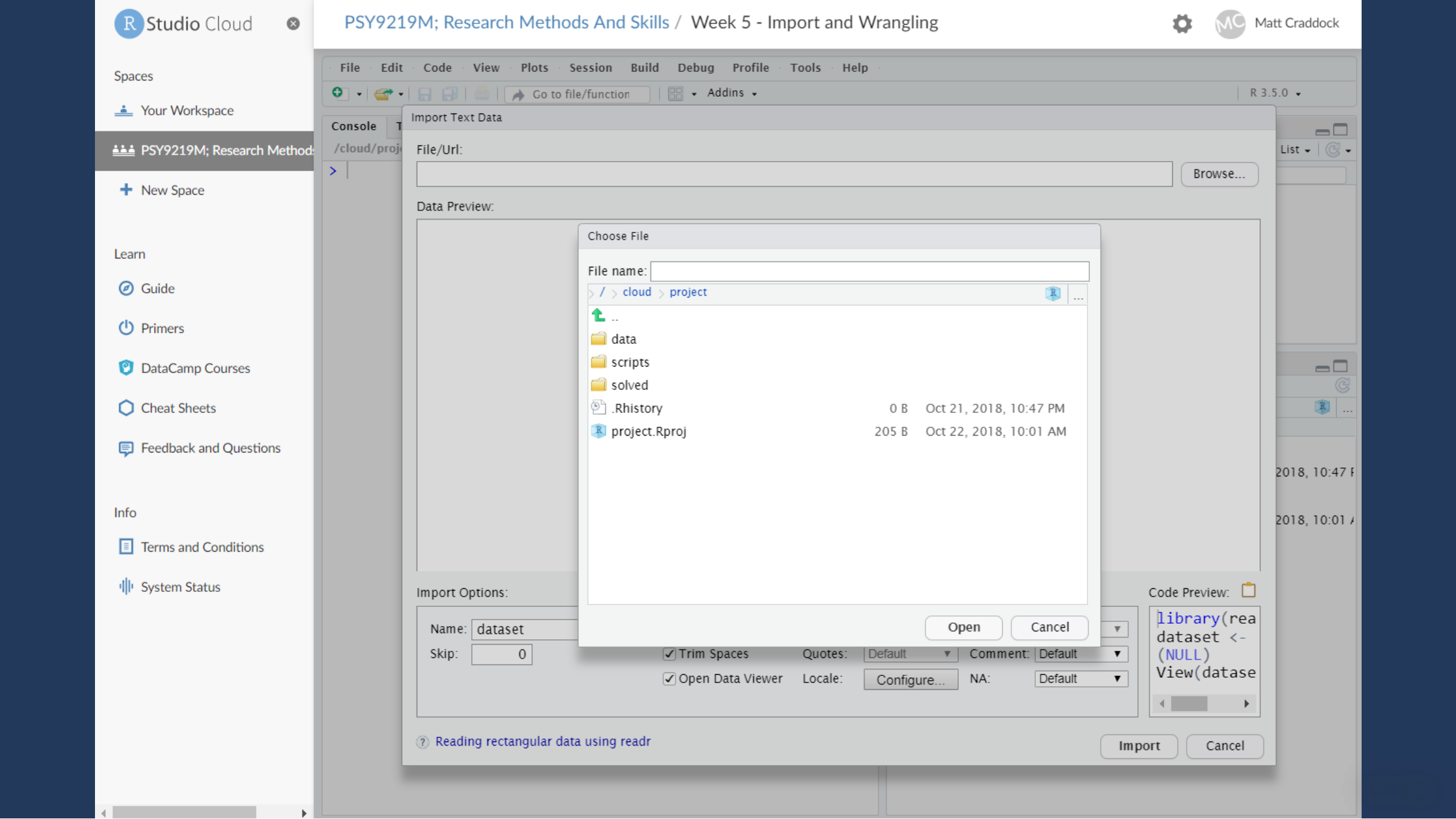
Each row represent a single participant's responses, and their demographic information.

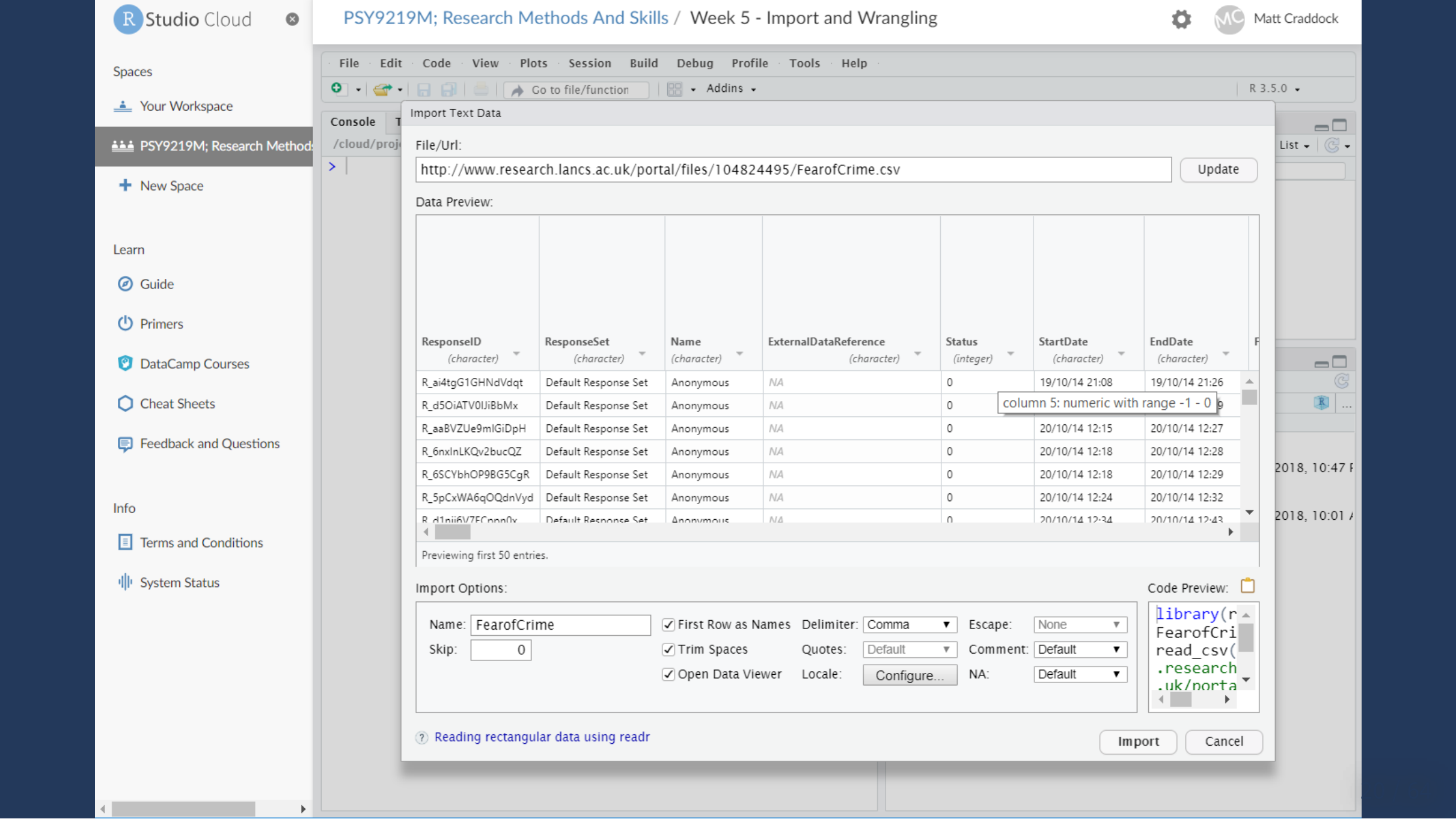
```
FearofCrime
```

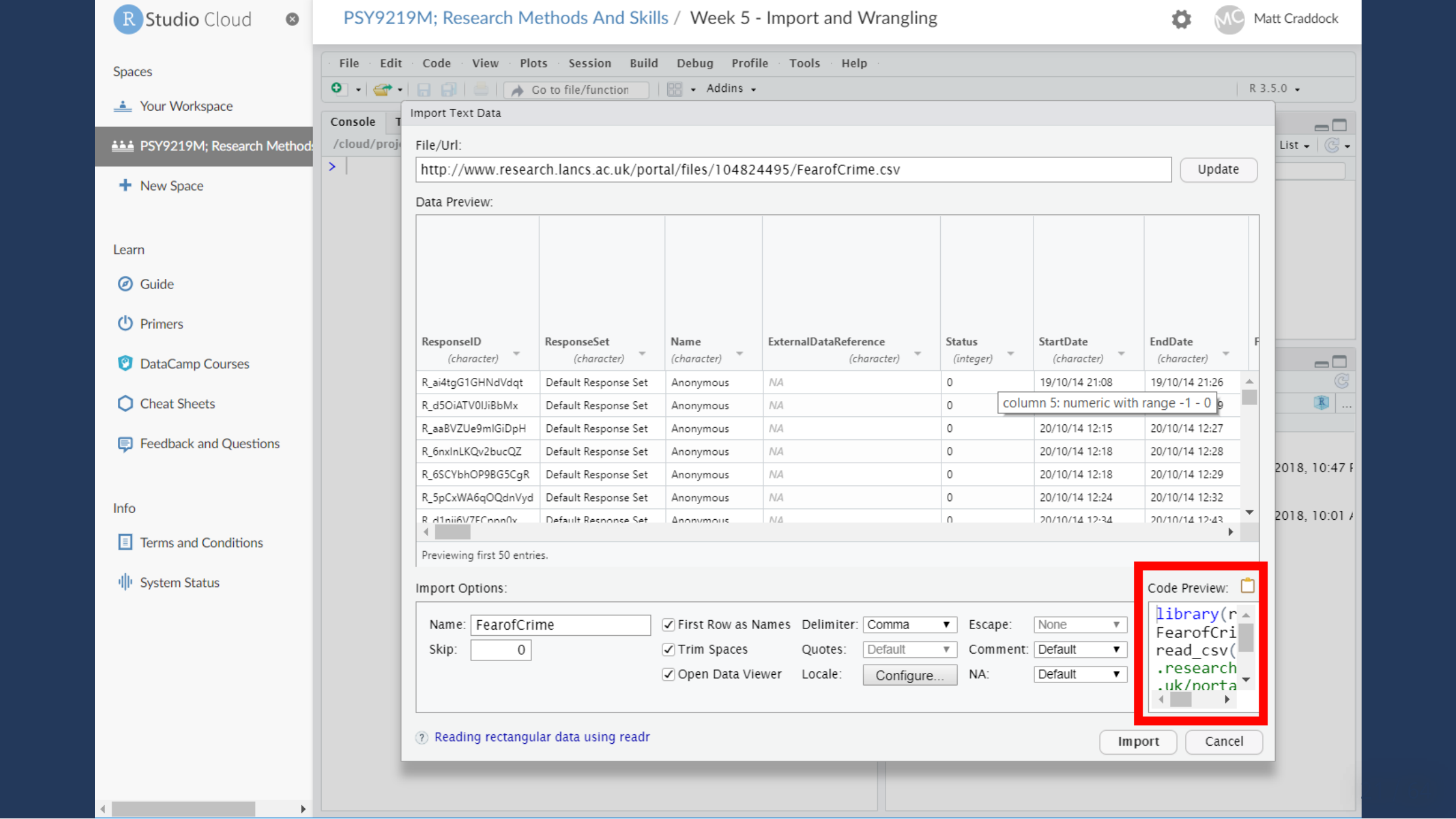
```
## # A tibble: 301 x 169
##    ResponseID  ResponseSet Name  ExternalDataRef~ Status StartDate EndDate Finished
##    <chr>       <chr>       <chr> <lgl>             <dbl> <chr>     <chr>      <dbl>
##  1 R_ai4tgG1G~ Default Re~ Anon~ NA                    0 19/10/14~ 19/10/~        1
##  2 R_d5OiATV0~ Default Re~ Anon~ NA                    0 20/10/14~ 20/10/~        1
##  3 R_aaBVZUe9~ Default Re~ Anon~ NA                    0 20/10/14~ 20/10/~        1
##  4 R_6nxInLKQ~ Default Re~ Anon~ NA                    0 20/10/14~ 20/10/~        1
##  5 R_6SCYbhOP~ Default Re~ Anon~ NA                    0 20/10/14~ 20/10/~        1
##  6 R_5pCxWA6q~ Default Re~ Anon~ NA                    0 20/10/14~ 20/10/~        1
##  7 R_d1nji6V7~ Default Re~ Anon~ NA                    0 20/10/14~ 20/10/~        1
##  8 R_9v6ZgUhK~ Default Re~ Anon~ NA                    0 20/10/14~ 20/10/~        1
##  9 R_5Bg7VjBh~ Default Re~ Anon~ NA                    0 20/10/14~ 20/10/~        1
## 10 R_9Sv17lQG~ Default Re~ Anon~ NA                    0 20/10/14~ 20/10/~        1
## # ... with 291 more rows, and 161 more variables: ...
```

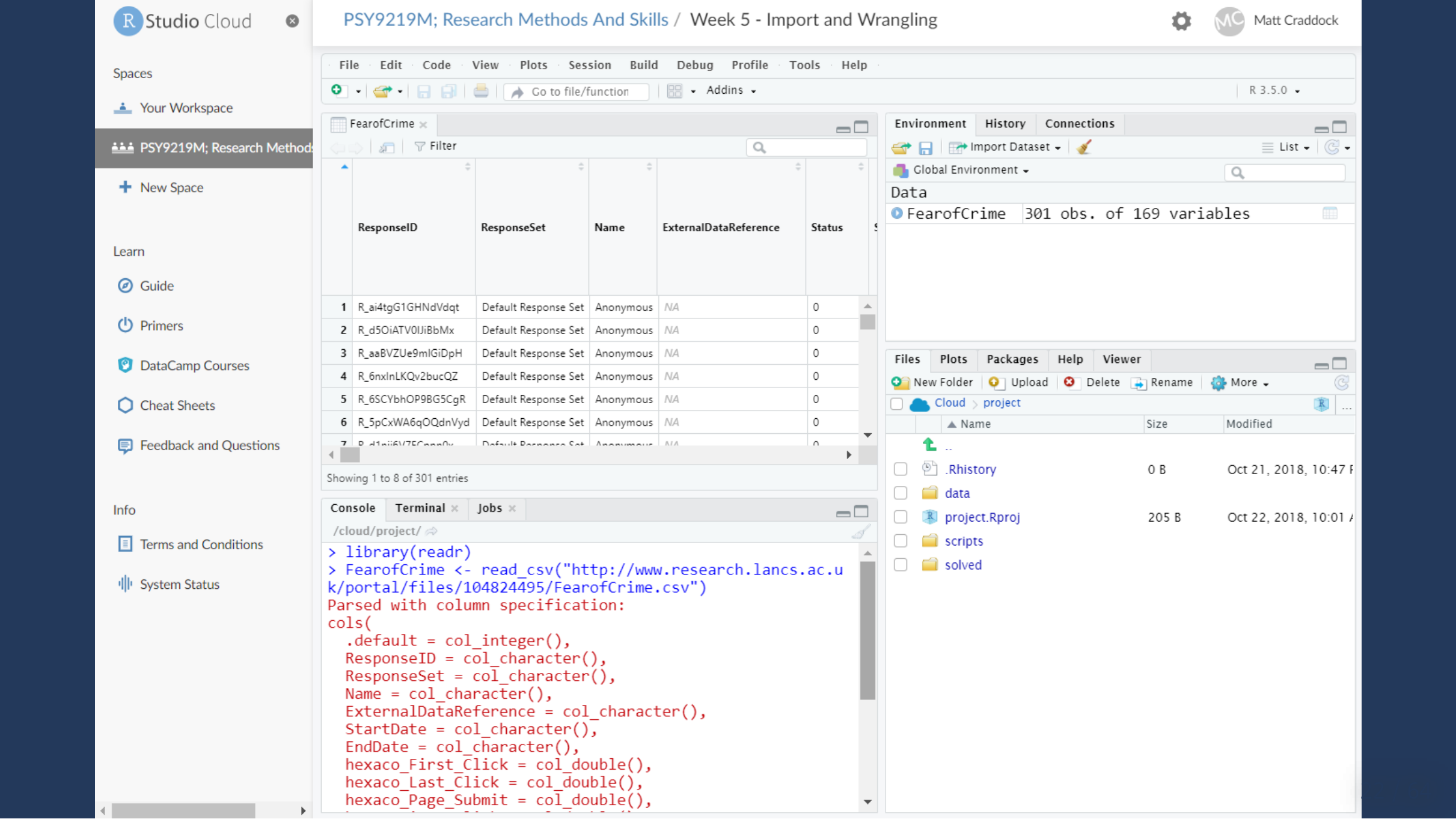

Studio Cloud
PSY9219M; Research Methods And Skills / Week 5 - Import and Wrangling
Matt Craddock
Spaces
Your Workspace
PSY9219M; Research Methods
New Space
Learn
Guide
Primers
DataCamp Courses
Cheat Sheets
Feedback and Questions
Info
Terms and Conditions
System Status
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
R 3.5.0
Console
Terminal
Jobs
/cloud/project/
Environment
History
Connections
Import Dataset
List
Global Environment
Environment is empty
Files
Plots
Packages
Help
Viewer
New Folder
Upload
Delete
Rename
More
Cloud
project
Name
Size
Modified
..
.Rhistory
0 B
Oct 21, 2018, 10:47
data
project.Rproj
205 B
Oct 22, 2018, 10:01
scripts
solved


Studio Cloud
PSY9219M; Research Methods And Skills / Week 5 - Import and Wrangling
Matt Craddock
Spaces
Your Workspace
PSY9219M; Research Methods
New Space
Learn
Guide
Primers
DataCamp Courses
Cheat Sheets
Feedback and Questions
Info
Terms and Conditions
System Status
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
R 3.5.0
Console
/cloud/proj
Import Text Data
File/Url:
Browse...
Data Preview:
Import Options:
Name: dataset
Skip: 0
First Row as Names
Trim Spaces
Open Data Viewer
Delimiter: Comma
Quotes: Default
Locale: Configure...
Escape: None
Comment: Default
NA: Default
Code Preview:
library(rea
dataset <-
(NULL)
View(datase
Reading rectangular data using readr
Import
Cancel
List
2018, 10:47
2018, 10:01

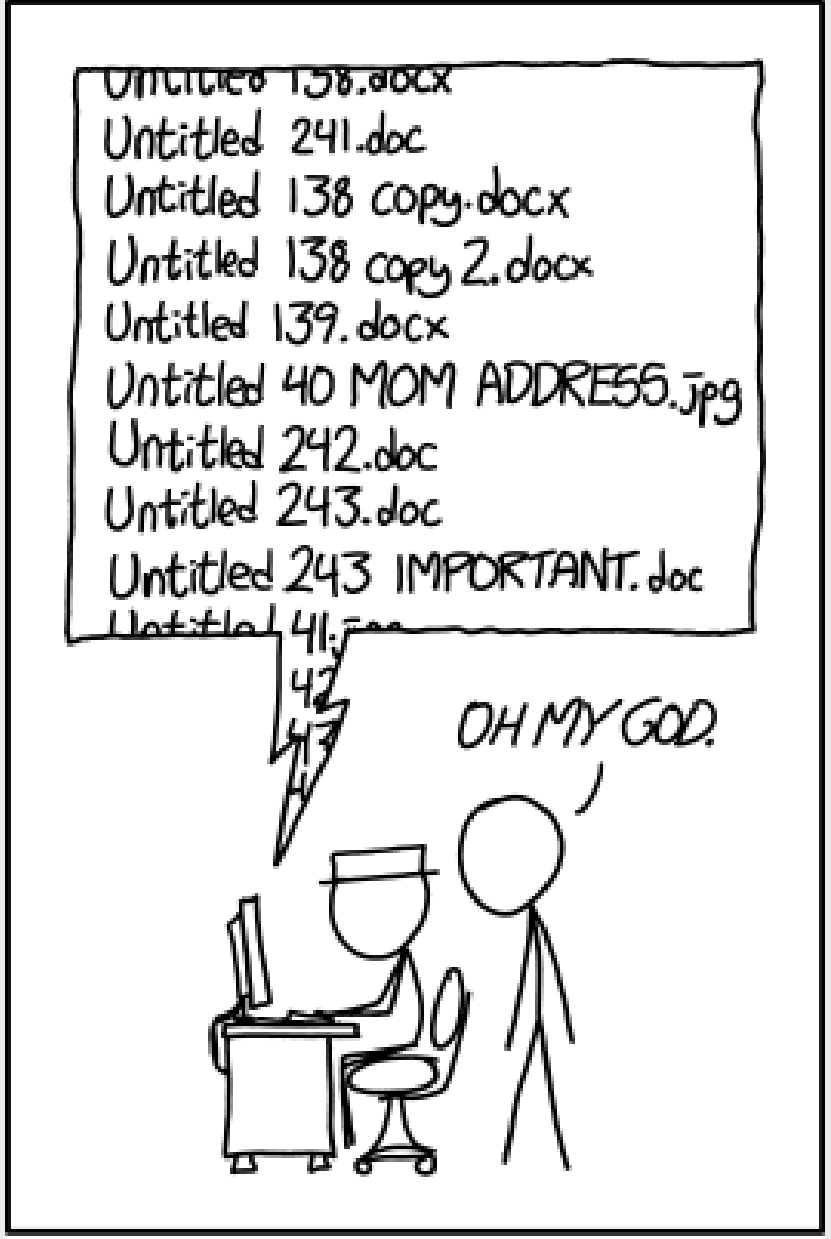

Studio Cloud
PSY9219M; Research Methods And Skills / Week 5 - Import and Wrangling
Matt Craddock
Spaces
Your Workspace
PSY9219M; Research Methods
New Space
Learn
Guide
Primers
DataCamp Courses
Cheat Sheets
Feedback and Questions
Info
Terms and Conditions
System Status
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
R 3.5.0
Console
Import Text Data
File/Url:
Browse...
Data Preview:
Choose File
File name:
/ cloud project
..
data
scripts
solved
.Rhistory 0 B Oct 21, 2018, 10:47 PM
project.Rproj 205 B Oct 22, 2018, 10:01 AM
Open
Cancel
Import Options:
Name: dataset
Skip: 0
Trim Spaces
Open Data Viewer
Quotes: Default
Locale: Configure...
Comment: Default
NA: Default
Code Preview:
library(rea
dataset <-
(NULL)
View(datase
Reading rectangular data using readr
Import
Cancel


Studio Cloud
PSY9219M; Research Methods And Skills / Week 5 - Import and Wrangling
Matt Craddock
Spaces
Your Workspace
PSY9219M; Research Methods
New Space
Learn
Guide
Primers
DataCamp Courses
Cheat Sheets
Feedback and Questions
Info
Terms and Conditions
System Status
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
R 3.5.0
Console
Import Text Data
File/Url:
http://www.research.lancs.ac.uk/portal/files/104824495/FearofCrime.csv
Update
Data Preview:
ResponseID (character)
ResponseSet (character)
Name (character)
ExternalDataReference (character)
Status (integer)
StartDate (character)
EndDate (character)
R_ai4tgG1GHNdVdqt Default Response Set Anonymous NA 0 19/10/14 21:08 19/10/14 21:26
R_d5OiATV0IJiBbMx Default Response Set Anonymous NA 0
column 5: numeric with range -1 - 0
R_aaBVZUe9mlGiDpH Default Response Set Anonymous NA 0 20/10/14 12:15 20/10/14 12:27
R_6nxInLKQv2bucQZ Default Response Set Anonymous NA 0 20/10/14 12:18 20/10/14 12:28
R_6SCYbhOP9BG5CgR Default Response Set Anonymous NA 0 20/10/14 12:18 20/10/14 12:29
R_5pCxWA6qOQdnVyd Default Response Set Anonymous NA 0 20/10/14 12:24 20/10/14 12:32
Previewing first 50 entries.
Import Options:
Name: FearofCrime
Skip: 0
First Row as Names
Trim Spaces
Open Data Viewer
Delimiter: Comma
Quotes: Default
Locale: Configure...
Escape: None
Comment: Default
NA: Default
Code Preview:
library(r
FearofCri
read_csv(
.research
Reading rectangular data using readr
Import
Cancel


Studio Cloud
PSY9219M; Research Methods And Skills / Week 5 - Import and Wrangling
Matt Craddock
Spaces
Your Workspace
PSY9219M; Research Methods
New Space
Learn
Guide
Primers
DataCamp Courses
Cheat Sheets
Feedback and Questions
Info
Terms and Conditions
System Status
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
R 3.5.0
Console
Import Text Data
File/Url:
http://www.research.lancs.ac.uk/portal/files/104824495/FearofCrime.csv
Update
Data Preview:
ResponseID (character)
ResponseSet (character)
Name (character)
ExternalDataReference (character)
Status (integer)
StartDate (character)
EndDate (character)
R_ai4tgG1GHNdVdqt Default Response Set Anonymous NA 0 19/10/14 21:08 19/10/14 21:26
R_d5OiATV0IJiBbMx Default Response Set Anonymous NA 0
column 5: numeric with range -1 - 0
R_aaBVZUe9mlGiDpH Default Response Set Anonymous NA 0 20/10/14 12:15 20/10/14 12:27
R_6nxInLKQv2bucQZ Default Response Set Anonymous NA 0 20/10/14 12:18 20/10/14 12:28
R_6SCYbhOP9BG5CgR Default Response Set Anonymous NA 0 20/10/14 12:18 20/10/14 12:29
R_5pCxWA6qOQdnVyd Default Response Set Anonymous NA 0 20/10/14 12:24 20/10/14 12:32
Previewing first 50 entries.
Import Options:
Name: FearofCrime
Skip: 0
First Row as Names
Trim Spaces
Open Data Viewer
Delimiter: Comma
Quotes: Default
Locale: Configure...
Escape: None
Comment: Default
NA: Default
Code Preview:
library(r
FearofCri
read_csv(
.research
.uk/porta
Reading rectangular data using readr
Import
Cancel


Studio Cloud
Spaces
Your Workspace
PSY9219M; Research Methods
New Space
Learn
Guide
Primers
DataCamp Courses
Cheat Sheets
Feedback and Questions
Info
Terms and Conditions
System Status
PSY9219M; Research Methods And Skills / Week 5 - Import and Wrangling
Matt Craddock
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
R 3.5.0
FearofCrime
Filter
ResponseID
ResponseSet
Name
ExternalDataReference
Status
1 R_ai4tgG1GHNdVdqt Default Response Set Anonymous NA 0
2 R_d5OiATV0IJiBbMx Default Response Set Anonymous NA 0
3 R_aaBVZUe9mlGiDpH Default Response Set Anonymous NA 0
4 R_6nxlnLKQv2bucQZ Default Response Set Anonymous NA 0
5 R_6SCYbhOP9BG5CgR Default Response Set Anonymous NA 0
6 R_5pCxWA6qOQdnVyd Default Response Set Anonymous NA 0
Showing 1 to 8 of 301 entries
Console
Terminal
Jobs
/cloud/project/
> library(readr)
> FearofCrime <- read_csv("http://www.research.lancs.ac.uk/portal/files/104824495/FearofCrime.csv")
Parsed with column specification:
cols(
.default = col_integer(),
ResponseID = col_character(),
ResponseSet = col_character(),
Name = col_character(),
ExternalDataReference = col_character(),
StartDate = col_character(),
EndDate = col_character(),
hexaco_First_Click = col_double(),
hexaco_Last_Click = col_double(),
hexaco_Page_Submit = col_double(),
Environment
History
Connections
Import Dataset
List
Global Environment
Data
FearofCrime 301 obs. of 169 variables
Files
Plots
Packages
Help
Viewer
New Folder
Upload
Delete
Rename
More
Cloud
project
Name
Size
Modified
.Rhistory 0 B Oct 21, 2018, 10:47
data
project.Rproj 205 B Oct 22, 2018, 10:01
scripts
solved

Keeping your analyses organised



PROTIP: NEVER LOOK IN SOMEONE ELSE'S DOCUMENTS FOLDER.

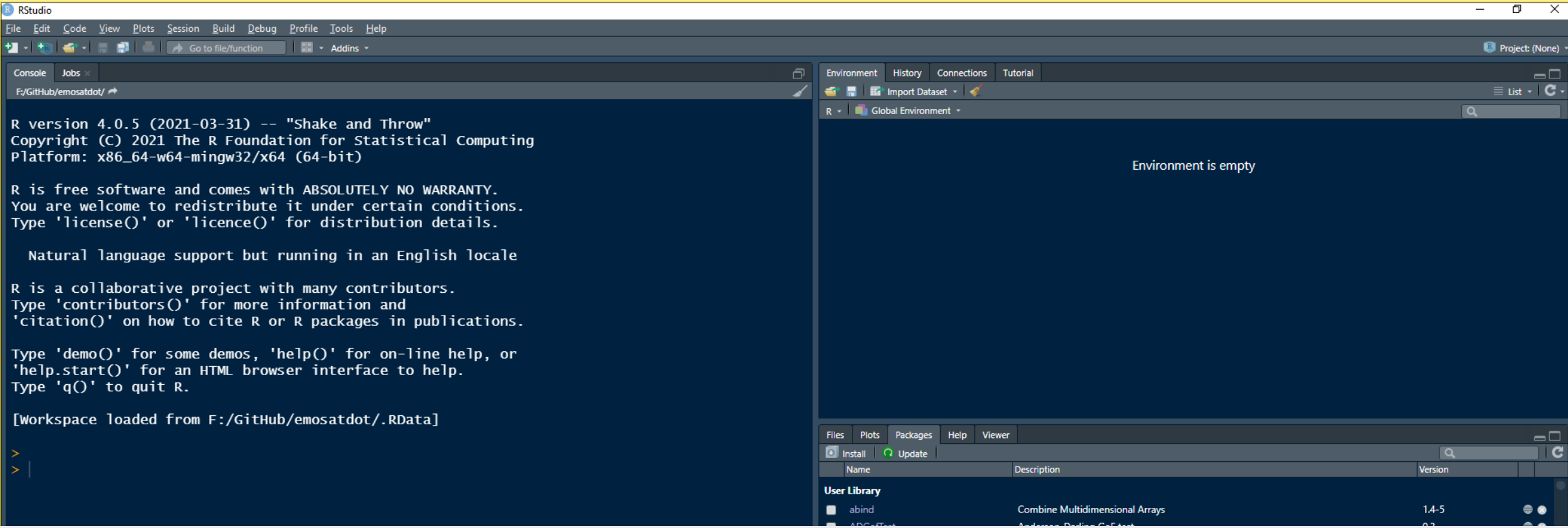
RStudio Projects

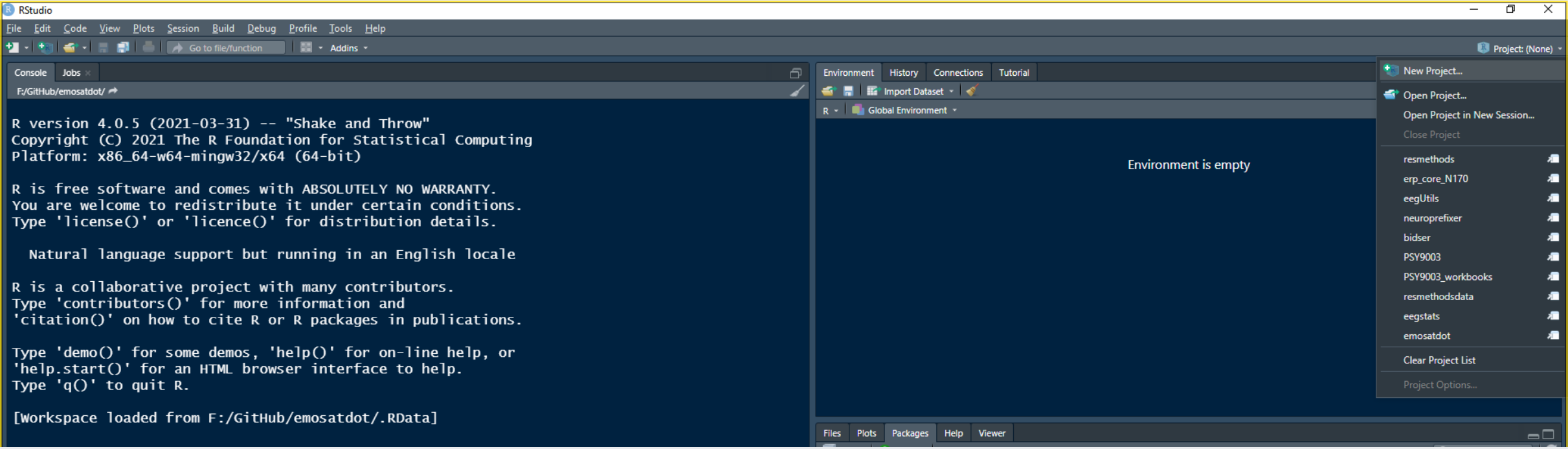
On RStudio.cloud, each project you create is in fact a completely separate instance of R.

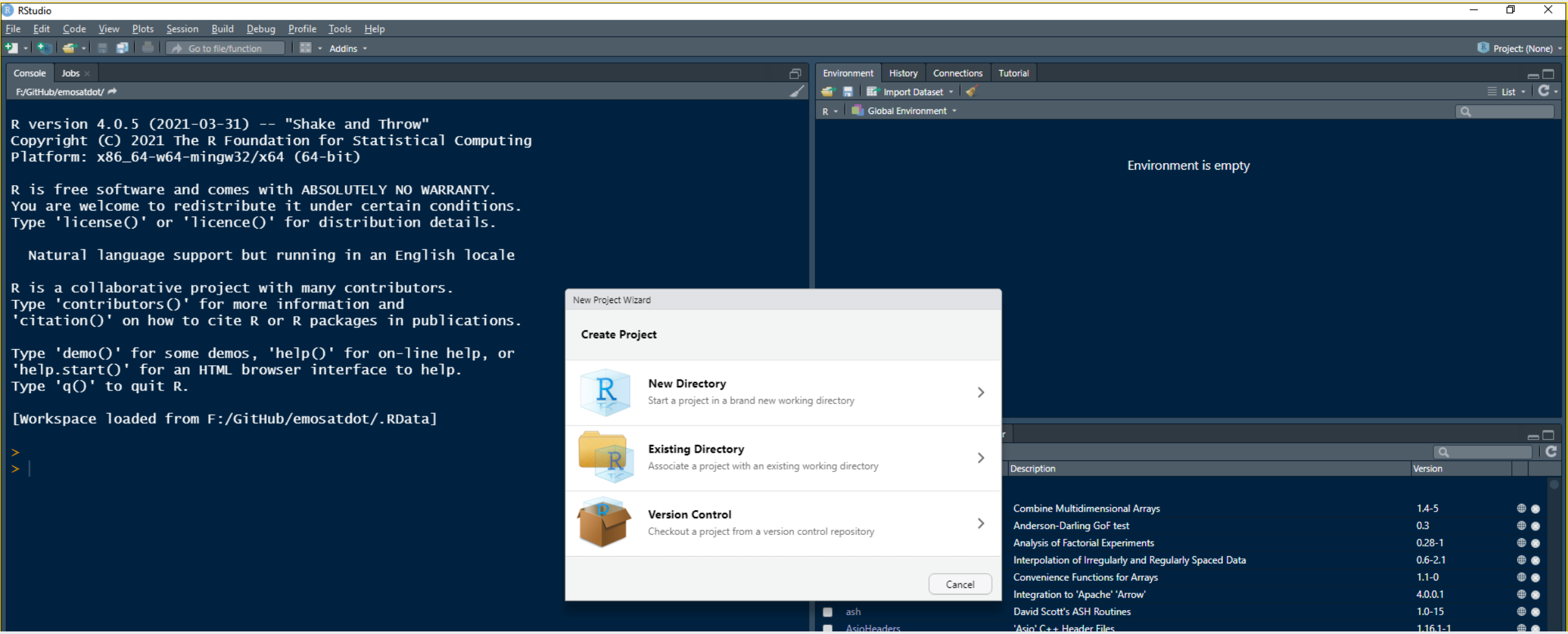
By now, most of you should have RStudio Desktop installed.

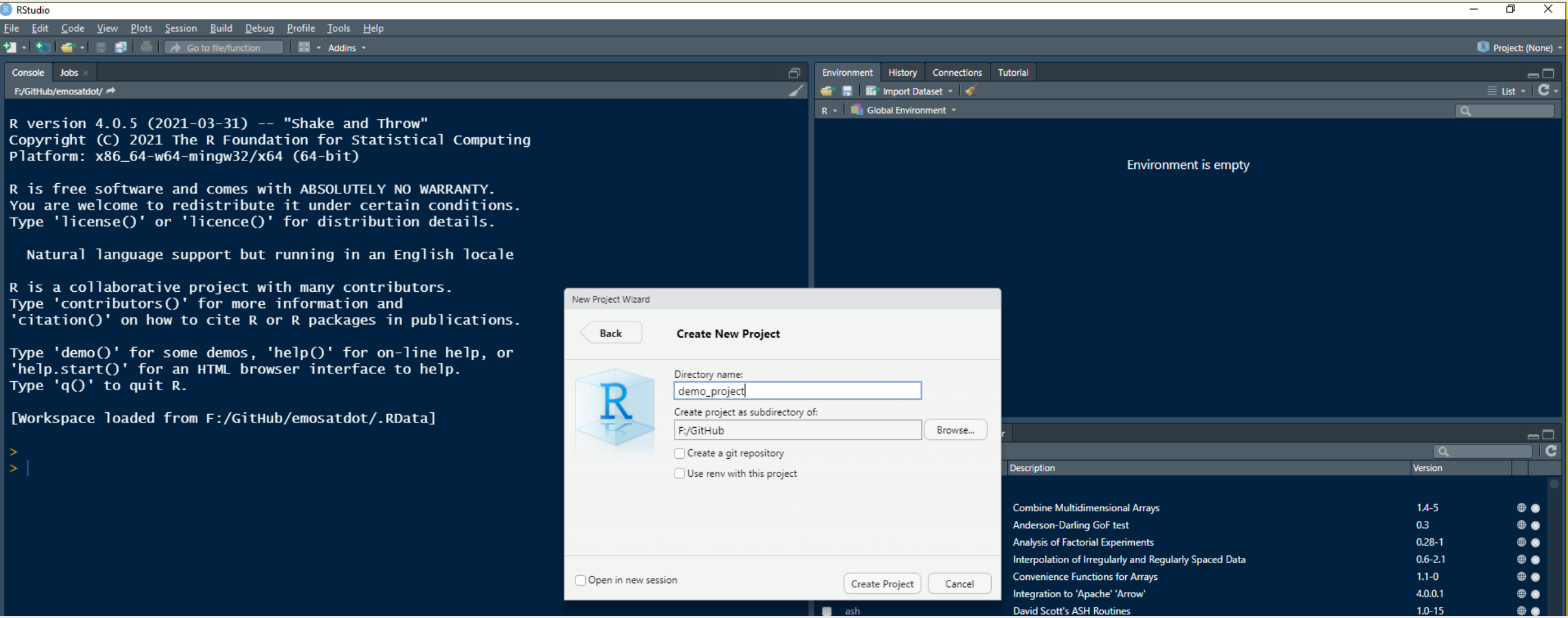
Once that's up and running, you can get to grips with RStudio projects

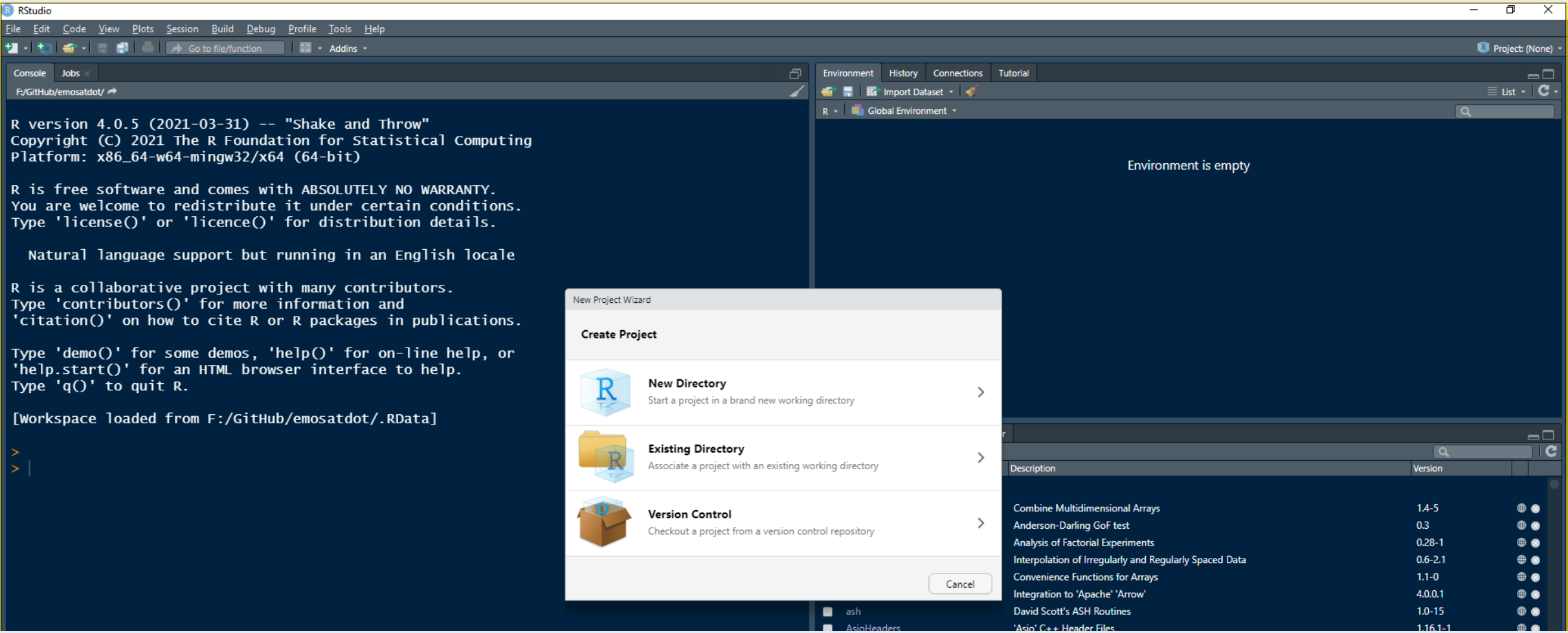
Projects provide a nice way to organise your work into neat, individually tailored sets of directories.


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Console
Jobs
F:/GitHub/emosatdot/
R version 4.0.5 (2021-03-31) -- "Shake and Throw"
Copyright (C) 2021 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
Natural language support but running in an English locale
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
[Workspace loaded from F:/GitHub/emosatdot/.RData]
Environment
History
Connections
Tutorial
Import Dataset
List
R
Global Environment
Environment is empty
Files
Plots
Packages
Help
Viewer
Install
Update
Name
Description
Version
User Library
abind
Combine Multidimensional Arrays
1.4-5


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Console
Jobs
F:/GitHub/emosatdot/
R version 4.0.5 (2021-03-31) -- "Shake and Throw"
Copyright (C) 2021 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
Natural language support but running in an English locale
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
[Workspace loaded from F:/GitHub/emosatdot/.RData]
Environment
History
Connections
Tutorial
Import Dataset
R
Global Environment
Environment is empty
Files
Plots
Packages
Help
Viewer
New Project...
Open Project...
Open Project in New Session...
Close Project
resmethods
erp_core_N170
eegUtils
neuroprefixer
bidser
PSY9003
PSY9003_workbooks
resmethodsdata
eegstats
emosatdot
Clear Project List
Project Options...


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Console
Jobs
F:/GitHub/emosatdot/
R version 4.0.5 (2021-03-31) -- "Shake and Throw"
Copyright (C) 2021 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
Natural language support but running in an English locale
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
[Workspace loaded from F:/GitHub/emosatdot/.RData]
Environment
History
Connections
Tutorial
Import Dataset
List
R
Global Environment
Environment is empty
New Project Wizard
Create Project
New Directory
Start a project in a brand new working directory
Existing Directory
Associate a project with an existing working directory
Version Control
Checkout a project from a version control repository
Cancel
Description
Version
Combine Multidimensional Arrays 1.4-5
Anderson-Darling GoF test 0.3
Analysis of Factorial Experiments 0.28-1
Interpolation of Irregularly and Regularly Spaced Data 0.6-2.1
Convenience Functions for Arrays 1.1-0
Integration to 'Apache' 'Arrow' 4.0.0.1
David Scott's ASH Routines 1.0-15
ash
AsioHeaders


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Console
Jobs
F:/GitHub/emosatdot/
R version 4.0.5 (2021-03-31) -- "Shake and Throw"
Copyright (C) 2021 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
Natural language support but running in an English locale
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
[Workspace loaded from F:/GitHub/emosatdot/.RData]
Environment
History
Connections
Tutorial
Import Dataset
List
R
Global Environment
Environment is empty
New Project Wizard
Back
Create New Project
Directory name:
demo_project
Create project as subdirectory of:
F:/GitHub
Browse...
Create a git repository
Use renv with this project
Open in new session
Create Project
Cancel
Description
Version
Combine Multidimensional Arrays 1.4-5
Anderson-Darling GoF test 0.3
Analysis of Factorial Experiments 0.28-1
Interpolation of Irregularly and Regularly Spaced Data 0.6-2.1
Convenience Functions for Arrays 1.1-0
Integration to 'Apache' 'Arrow' 4.0.0.1
David Scott's ASH Routines 1.0-15


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Console
Jobs
F:/GitHub/emosatdot/
R version 4.0.5 (2021-03-31) -- "Shake and Throw"
Copyright (C) 2021 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
Natural language support but running in an English locale
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
[Workspace loaded from F:/GitHub/emosatdot/.RData]
Environment
History
Connections
Tutorial
Import Dataset
List
R
Global Environment
Environment is empty
New Project Wizard
Create Project
New Directory
Start a project in a brand new working directory
Existing Directory
Associate a project with an existing working directory
Version Control
Checkout a project from a version control repository
Cancel
Description
Version
Combine Multidimensional Arrays 1.4-5
Anderson-Darling GoF test 0.3
Analysis of Factorial Experiments 0.28-1
Interpolation of Irregularly and Regularly Spaced Data 0.6-2.1
Convenience Functions for Arrays 1.1-0
Integration to 'Apache' 'Arrow' 4.0.0.1
David Scott's ASH Routines 1.0-15
ash

Keeping your analyses organised

Make a new RStudio project for each week's exercises!

Follow sensible structure:

Keep your data in a folder called data.

Keep your scripts or RMarkdown documents in a folder called scripts.

Give your files sensible names!

For more general workflow advice, check out What They Forgot to Teach You About R at https://rstats.wtf/

Relating data to structure

Let's think about an *experiment*

The experiment is a reaction time experiment with a two-by-two repeated measures design.

Participants see pictures of objects twice. Sometimes they are seen from the *same* viewpoint twice, sometimes from *different* viewpoints each time.

There are two separate blocks of trials. The dependent variable is how long it takes them to name the objects, or *reaction time*.

You're interested in whether:

1. they get faster at naming object the second time

2. they are faster when the same view is presented both times.

How many variables are there?

Variables	R Data Type
Participant ID	Numeric or character
Reaction times	Numeric
Block first/second	Character/factor
Viewpoint same/different	Character/factor

The final dataset needs to be able to do several things.

1. It needs to uniquely identify each participant.
2. It needs to tie each value to the right participant.
3. It needs to identify what each value represents in terms of the design.

Tidy data

The three principles of tidy data

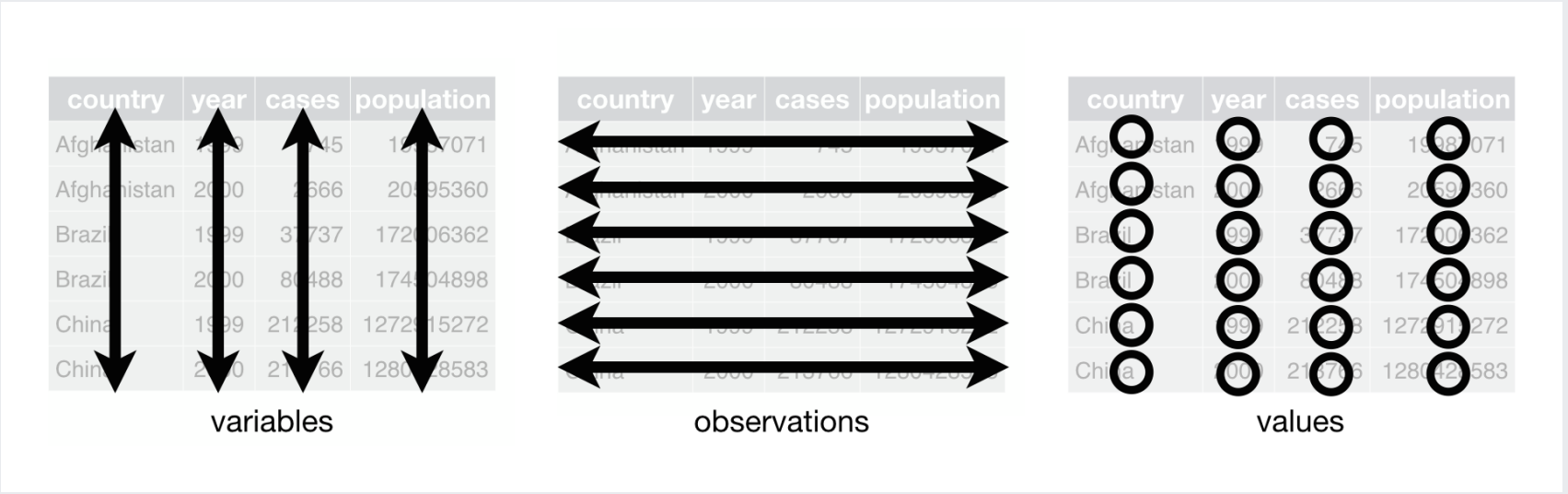
1. Each variable must have its own column.
2. Each observation must have its own row.
3. Each value must have its own cell.

Why Tidy?

1. Many functions in R operate on so-called *long* format data, requiring dependent and independent variables to be in different columns of a data frame.

2. Having a consistent way to store and structure your data makes it more *generic*. This makes it easier to use it with different functions.

3. Being *generic* also makes it easier to understand a new dataset in this format.



The many ways to structure data

One column for condition, one column for RT

```
## # A tibble: 40 x 3
## # Groups:   Participant [10]
##    Participant exp_condition       RT
##          <int> <chr>            <dbl>
##  1           1 Block1_different  407.
##  2           1 Block1_same       415.
##  3           1 Block2_different  382.
##  4           1 Block2_same       371.
##  5           2 Block1_different  420.
##  6           2 Block1_same       384.
##  7           2 Block2_different  479.
##  8           2 Block2_same       402.
##  9           3 Block1_different  368.
## 10           3 Block1_same       341.
## # ... with 30 more rows
```

This is a little awkward.

At first glance, there's no easy way to see how many variables there.

Dependent variable split across columns

```
## # A tibble: 16 x 4
## # Groups:   Participant [8]
##    Participant Viewpoint  B1RT  B2RT
##          <int> <chr>     <dbl> <dbl>
##  1           1 Different  536.  364.
##  2           1 Same       494.  450.
##  3           2 Different  511.  393.
##  4           2 Same       432.  371.
##  5           3 Different  536.  364.
##  6           3 Same       494.  450.
##  7           4 Different  511.  393.
##  8           4 Same       432.  371.
##  9           5 Different  536.  364.
## 10           5 Same       494.  450.
## 11           6 Different  511.  393.
## 12           6 Same       432.  371.
## 13           7 Different  536.  364.
## 14           7 Same       494.  450.
## 15           8 Different  511.  393.
## 16           8 Same       432.  371.
```

Now there's a mishmash of things:

One variable (Viewpoint) is in one column.

The Block variable is spread across two columns.

The dependent variable (reaction time) is spread across two columns.

One column per condition

```
## # A tibble: 10 x 5
##    Participant Block1_same Block2_same Block1_different Block2_different
##          <int>       <dbl>       <dbl>             <dbl>             <dbl>
##  1           1        515.        268.              546.              413.
##  2           2        471.        249.              535.              449.
##  3           3        507.        331.              501.              386.
##  4           4        482.        312.              607.              389.
##  5           5        484.        322.              595.              431.
##  6           6        502.        301.              527.              359.
##  7           7        520.        328.              557.              398.
##  8           8        579.        272.              578.              378.
##  9           9        441.        290.              572.              401.
## 10          10        526.        285.              550.              405.
```

This is also called **wide** format.

How many *variables* are there?

```
## # A tibble: 10 x 5
##    Participant Block1_same Block2_same Block1_different Block2_different
##          <int>       <dbl>       <dbl>             <dbl>             <dbl>
##  1           1        515.        268.              546.              413.
##  2           2        471.        249.              535.              449.
##  3           3        507.        331.              501.              386.
##  4           4        482.        312.              607.              389.
##  5           5        484.        322.              595.              431.
##  6           6        502.        301.              527.              359.
##  7           7        520.        328.              557.              398.
##  8           8        579.        272.              578.              378.
##  9           9        441.        290.              572.              401.
## 10          10        526.        285.              550.              405.
```

Four... but there are five columns.

```
ncol(example_rt_df)
```

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

How many *observations* are there?

```
## # A tibble: 10 x 5
##    Participant Block1_same Block2_same Block1_different Block2_different
##          <int>       <dbl>       <dbl>             <dbl>             <dbl>
##  1           1        515.        268.              546.              413.
##  2           2        471.        249.              535.              449.
##  3           3        507.        331.              501.              386.
##  4           4        482.        312.              607.              389.
##  5           5        484.        322.              595.              431.
##  6           6        502.        301.              527.              359.
##  7           7        520.        328.              557.              398.
##  8           8        579.        272.              578.              378.
##  9           9        441.        290.              572.              401.
## 10          10        526.        285.              550.              405.
```

40... but there are 10 rows.

```
nrow(example_rt_df)
```

```
## [1] 10
```

One column per condition

```
## # A tibble: 10 x 5
##    Participant Block1_same Block2_same Block1_different Block2_different
##          <int>       <dbl>       <dbl>             <dbl>             <dbl>
##  1           1        515.        268.              546.              413.
##  2           2        471.        249.              535.              449.
##  3           3        507.        331.              501.              386.
##  4           4        482.        312.              607.              389.
##  5           5        484.        322.              595.              431.
##  6           6        502.        301.              527.              359.
##  7           7        520.        328.              557.              398.
##  8           8        579.        272.              578.              378.
##  9           9        441.        290.              572.              401.
## 10          10        526.        285.              550.              405.
```

This is also called **wide** format.

This data is *untidy*

One variable - RT - is split across four columns.

Another variable - Block - is split across two columns.

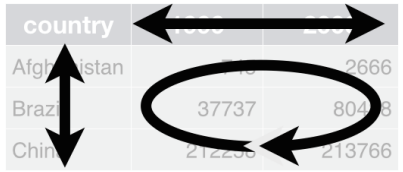
A third variable - viewpoint - is also split across two columns.

country	1999	2000
Afghanistan	745	2666
Brazil	37737	80488
China	212258	213766

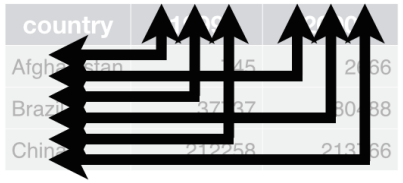
table4

country	1999	2000
Afghanistan	19987071	20595360
Brazil	172006362	174504898
China	1272915272	1280428583

table5



variables



observations

Tidying your data

Tidyr

The `tidyr` package contains functions to help tidy up your data.

We'll look now at `pivot_longer()` and `pivot_wider()`.

To start tidying our data, we need the RTs to be in a single column.

```
head(example_rt_df, n = 4)
```

```
## # A tibble: 4 x 5
##   Participant Block1_same Block2_same Block1_different Block2_different
##         <int>       <dbl>       <dbl>             <dbl>             <dbl>
## 1           1        508.        340.              522.              295.
## 2           2        523.        268.              550.              470.
## 3           3        543.        303.              667.              476.
## 4           4        556.        408.              400.              322.
```

The function `pivot_longer()` can be used to combine columns into one.

Look at the help using `?pivot_longer`

Pivoting longer

```
pivot_longer(data,
             cols,
             names_to = "key",
             values_to = "value",
             ...)
```

The first argument, `data`, is the name of the data frame you want to modify.

`cols` are the columns you want to combine together.

`names_to` is the name of the new column that will contain the values of a single categorical variable.

`values_to` is the name of the new column containing the values for each level of that variable.

Pivoting longer

```
long_rt <-
  pivot_longer(example_rt_df,
               cols = c("Block1_same",
                        "Block1_different",
                        "Block2_same",
                        "Block2_different"),
               names_to = "exp_cond",
               values_to = "RT")
head(long_rt)
```

```
## # A tibble: 6 x 3
##   Participant exp_cond            RT
##         <int> <chr>            <dbl>
## 1           1 Block1_same       508.
## 2           1 Block1_different  522.
## 3           1 Block2_same       340.
## 4           1 Block2_different  295.
## 5           2 Block1_same       523.
## 6           2 Block1_different  550.
```

Pivoting longer

After we specify the "key" and "value" columns, we need to specify which columns we want to be *gathered*.

```
long_rt <-
  pivot_longer(example_rt_df,
               names_to = "exp_cond",
               values_to = "RT",
               cols = 2:5) # here I use num
head(long_rt)
```

```
## # A tibble: 6 x 3
##   Participant exp_cond           RT
##         <int> <chr>             <dbl>
## 1           1 Block1_same       508.
## 2           1 Block2_same       340.
## 3           1 Block1_different  522.
## 4           1 Block2_different  295.
## 5           2 Block1_same       523.
## 6           2 Block2_same       268.
```

```
long_rt <-
  pivot_longer(example_rt_df,
               names_to = "exp_cond",
               values_to = "RT",
               cols = Block1_same:Block2_di
head(long_rt)
```

```
## # A tibble: 6 x 3
##   Participant exp_cond           RT
##         <int> <chr>             <dbl>
## 1           1 Block1_same       508.
## 2           1 Block2_same       340.
## 3           1 Block1_different  522.
## 4           1 Block2_different  295.
## 5           2 Block1_same       523.
## 6           2 Block2_same       268.
```

Splitting columns

We have the RTs in one column, but we still have another problem:

The "Block" and "Viewpoint" variables are combined into a single column.

```
head(long_rt)
```

```
## # A tibble: 6 x 3
##   Participant exp_cond            RT
##         <int> <chr>            <dbl>
## 1           1 Block1_same       508.
## 2           1 Block2_same       340.
## 3           1 Block1_different  522.
## 4           1 Block2_different  295.
## 5           2 Block1_same       523.
## 6           2 Block2_same       268.
```

Splitting columns

Fortunately, the values in the *exp_cond* column can be easily split:

```
unique(long_rt$exp_cond)
```

```
## [1] "Block1_same"      "Block2_same"      "Block1_different" "Block2_different"
```

The value of "Block" comes before the underscore ("_"), while the value of "viewpoint" comes after it.

Splitting columns

```
long_rt <-
  pivot_longer(example_rt_df,
               names_to = c("Block",
                            "Viewpoint"),
               names_sep = "_",
               values_to = "RT",
               cols = Block1_same:Block2_different)
head(long_rt)
```

```
## # A tibble: 6 x 4
##   Participant Block  Viewpoint    RT
##         <int> <chr>  <chr>     <dbl>
## 1           1 Block1 same       508.
## 2           1 Block2 same       340.
## 3           1 Block1 different  522.
## 4           1 Block2 different  295.
## 5           2 Block1 same       523.
## 6           2 Block2 same       268.
```

Splitting columns

Let's look at the additional syntax.

```
pivot_longer(example_rt_df,
             names_to = c("Block",
                          "Viewpoint"),
             names_sep = "_",
             values_to = "RT",
             cols = Block1_same:Block2_different)
```

`names_to` now has two entries, one for each new column that will be made.

`names_sep` is the character that *separates* the values you want to split.

Your target

```
## # A tibble: 15 x 4
##    Participant Block  Viewpoint    RT
##          <int> <chr>  <chr>     <dbl>
##  1           1 Block1 same       508.
##  2           1 Block2 same       340.
##  3           1 Block1 different  522.
##  4           1 Block2 different  295.
##  5           2 Block1 same       523.
##  6           2 Block2 same       268.
##  7           2 Block1 different  550.
##  8           2 Block2 different  470.
##  9           3 Block1 same       543.
## 10           3 Block2 same       303.
## 11           3 Block1 different  667.
## 12           3 Block2 different  476.
## 13           4 Block1 same       556.
## 14           4 Block2 same       408.
## 15           4 Block1 different  400.
```

You should specify name(s) for the column(s) that you'll create using the `names_to` and `values_to` arguments.

You'll need to add `names_sep` and the character that separates the two sides as well in order to match the target

Pivoting wider

Pivoting wider

Sometimes you want to go in the *opposite* direction.

`pivot_wider()` is the *opposite* of `pivot_longer()`.

```
wide_rt <-
  pivot_wider(long_rt,
              names_from = c("Block",
                             "Viewpoint"),
              values_from = "RT")
head(wide_rt, 10)
```

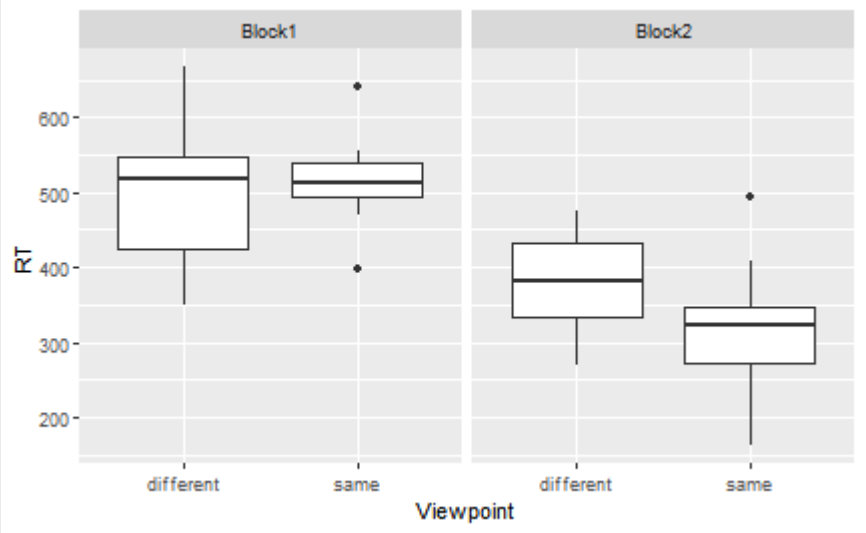
```
## # A tibble: 10 x 5
##    Participant Block1_same Block2_same Block1_different Block2_different
##          <int>       <dbl>       <dbl>             <dbl>             <dbl>
##  1           1        508.        340.              522.              295.
##  2           2        523.        268.              550.              470.
##  3           3        543.        303.              667.              476.
##  4           4        556.        408.              400.              322.
##  5           5        506.        163.              539.              269.
##  6           6        489.        287.              350.              363.
##  7           7        398.        346.              624.              392.
##  8           8        470.        494.              504.              374.
##  9           9        517.        258.              396.              422.
```

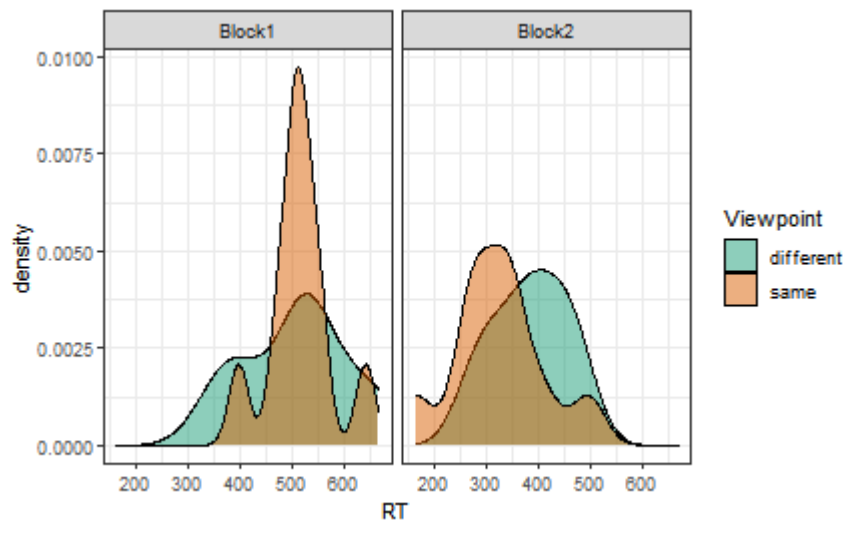
Now what?

Now that it's tidy...

Now that we've got the data in a tidy format, we can begin to use some of the more interesting features of R!

We can produce a boxplot using **ggplot2** (more next week!)





Now that it's tidy...

We can produce some summary statistics using **dplyr** (more soon!)

```
## # A tibble: 4 x 4
## # Groups:   Block [2]
##   Block  Viewpoint mean_RT sd_RT
##   <chr>  <chr>       <dbl> <dbl>
## 1 Block1 different    507. 100.
## 2 Block1 same         515.  62.5
## 3 Block2 different    382.  71.2
## 4 Block2 same         321.  89.7
```

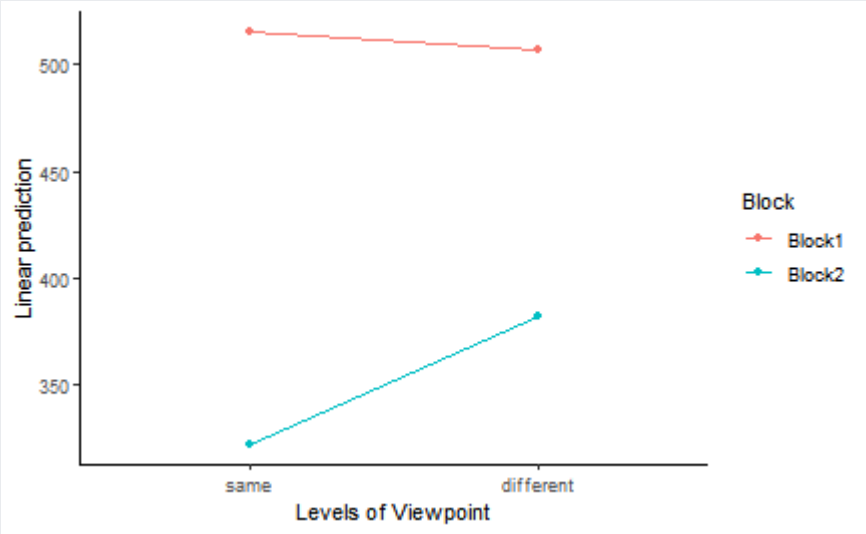
Now that it's tidy...

We can run ANOVA with **afex**.

```
## Anova Table (Type 3 tests)
##
## Response: RT
##            Effect   df     MSE         F  ges p.value
## 1           Block 1, 9 5222.72 48.56 *** .510   <.001
## 2       Viewpoint 1, 9 7794.41      0.87 .027    .376
## 3 Block:Viewpoint 1, 9 6343.29      1.87 .046    .205
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '+' 0.1 ' ' 1
```

Now that it's tidy...

We can create interaction plots using **emmeans**.




```
## Block = Block1:
##  contrast         estimate   SE df t.ratio p.val
##  same - different     8.43 39.9  9   0.212  0.83
##
## Block = Block2:
##  contrast         estimate   SE df t.ratio p.val
##  same - different   -60.45 35.2  9  -1.717  0.12
```

Next week

- The following chapters of R for Data Science -
  - **Data Visualization** (Chapter 1 via the library)
  - **Graphics for communication with ggplot2** (Chapter 22 via the library)

Practice some of the skills for next week:

- **RStudio.cloud Primer**
  - Visualize Data

A possible solution for the extra exercise!

```
set.seed(200) # if you want these exact numbers, use this line
example_rt_df <-
  tibble(Participant = seq(1, 10),
         Block1_same = rnorm(10, 500, 100),
         Block2_same = rnorm(10, 350, 100),
         Block1_different = rnorm(10, 500, 100),
         Block2_different = rnorm(10, 400, 100))
```

```
## # A tibble: 5 x 5
##   Participant Block1_same Block2_same Block1_different Block2_different
##         <int>       <dbl>       <dbl>             <dbl>             <dbl>
## 1           1        508.        340.              522.              295.
## 2           2        523.        268.              550.              470.
## 3           3        543.        303.              667.              476.
## 4           4        556.        408.              400.              322.
## 5           5        506.        163.              539.              269.
```