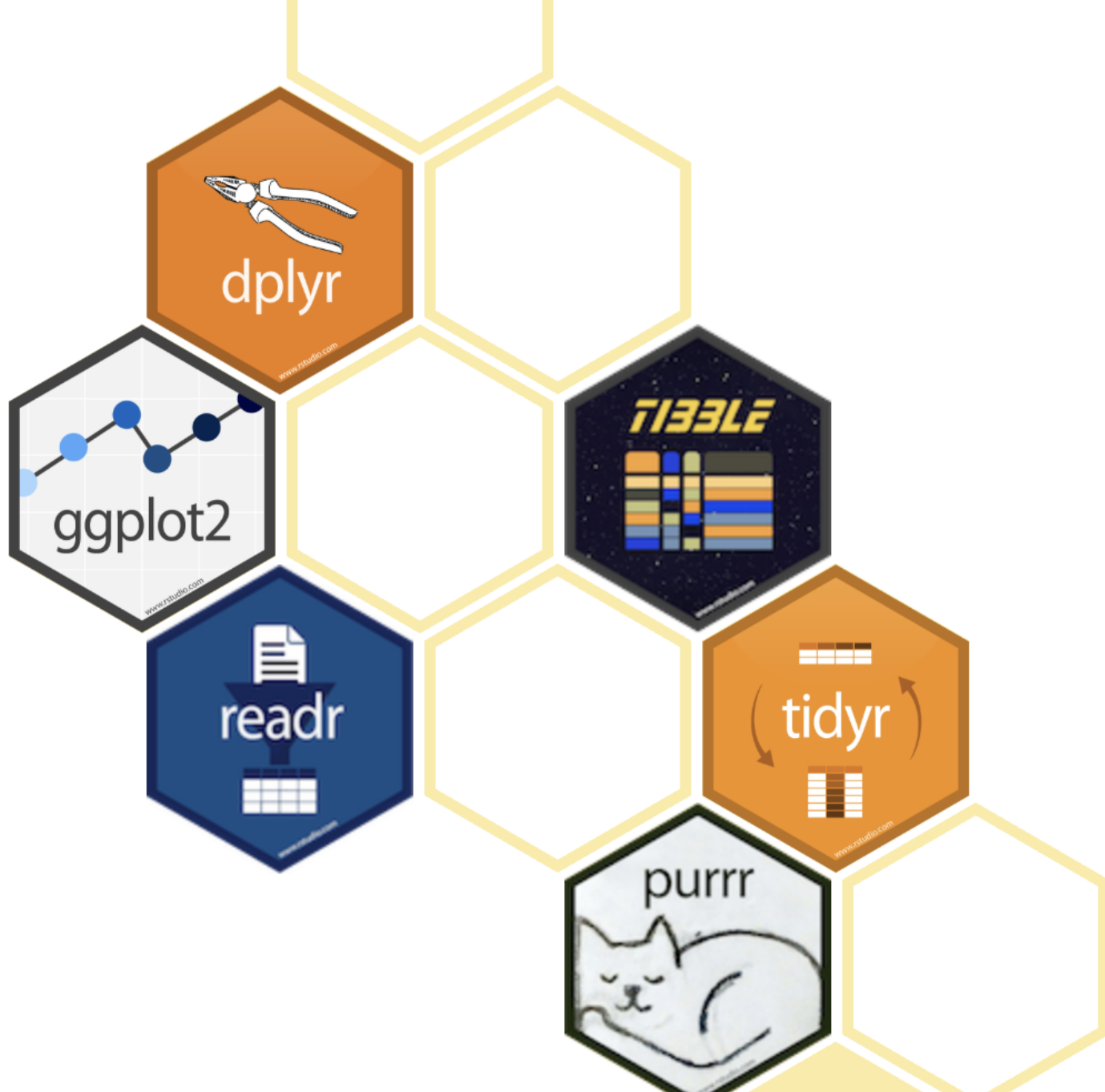


# Introduction to the Tidyverse

**Import, wrangle, model, and  
communicate data**

2020-09-22



# Working with data in R

**the tidyverse is a collection of friendly and consistent tools for data analysis and visualization.**

# Working with data in R

the tidyverse is a collection of friendly and consistent tools for data analysis and visualization.

**They live as, R packages, each of which does one thing well.**

# library(tidyverse) will load the core packages:

**ggplot2**, for data visualisation.

**dplyr**, for data manipulation.

**tidyr**, for data tidying.

**readr**, for data import.

**purrr**, for functional programming.

**tibble**, for tibbles, a modern re-imagining of data frames.

**stringr**, for strings.

**forcats**, for factors.



# exercises.Rmd

```
---
title: "Import Data"
output: html_document
---
```

```
```{r setup}
library(tidyverse)
library(haven)
```
```



In this section, we will learn about importing and exporting files from common file formats, including CSV and formats from other statistical software using the readr and haven packages.

## ## readr

readr supplies several related functions, each designed to read in a specific flat file format.

| Function       | Reads                      |
|----------------|----------------------------|
| -----          | -----                      |
| `read_csv()`   | Comma separated values     |
| `read_csv2()`  | Semi-colon separate values |
| `read_delim()` | General delimited files    |
| `read_fwf()`   | Fixed width files          |
| `read_log()`   | Apache log files           |

# readr ↕




# code chunks

```
```\r}  
csv_data <- read_csv(  
  "a,b,c,d  
  1,2,3,4  
  5,6,7,8",  
  col_types = ""  
)  
  
csv_data  
```\r
```

# running code chunks

```
```{r}
csv_data <- read_csv(
  "a,b,c,d
1,2,3,4
5,6,7,8",
  col_types = ""
)

csv_data|
```
```

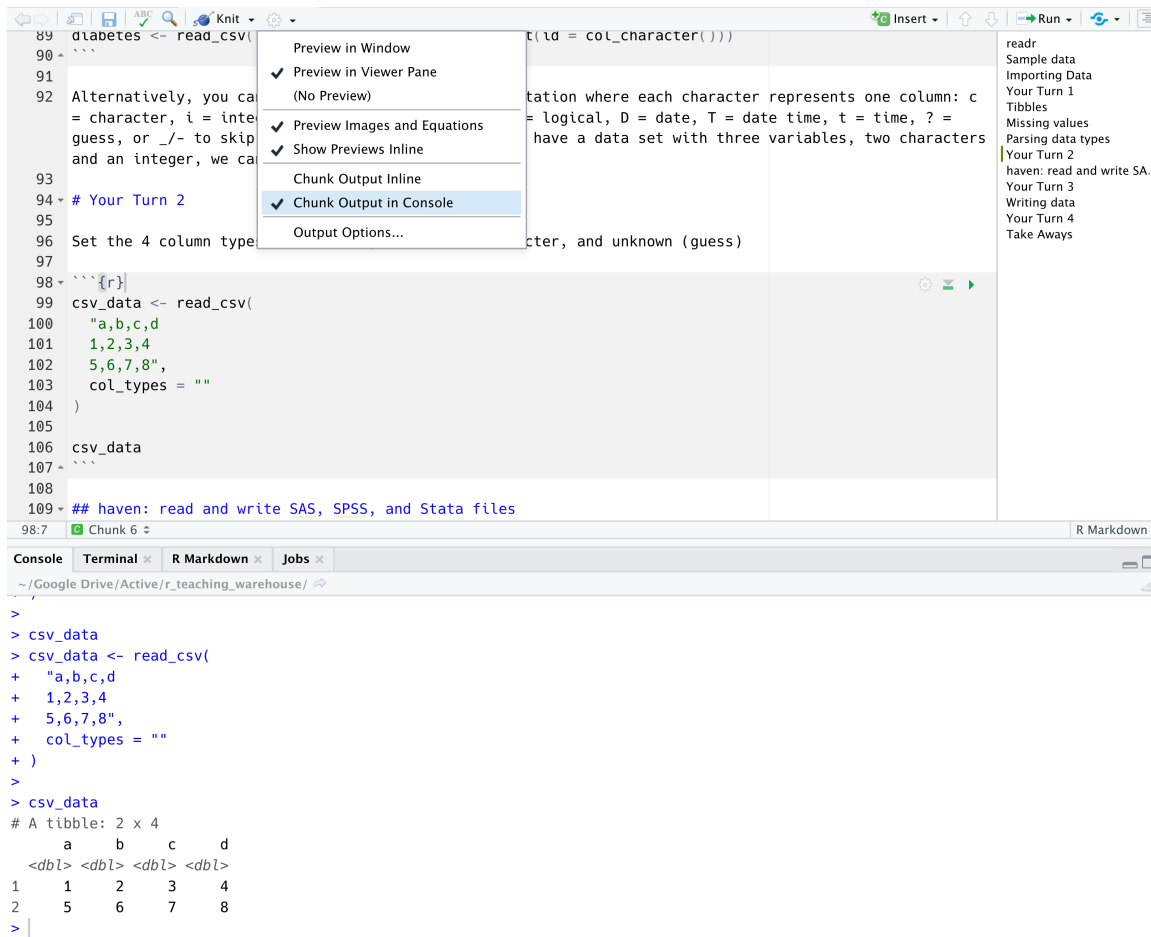


| <b>a</b> | <b>b</b> | <b>c</b> | <b>d</b> |
|----------|----------|----------|----------|
| <dbl>    | <dbl>    | <dbl>    | <dbl>    |
| 1        | 2        | 3        | 4        |
| 5        | 6        | 7        | 8        |

2 rows



# outputting to the console



```
89 diabetes <- read_csv(
90 ...
91
92 Alternatively, you can use the guess engine to guess the column types. For example,
93 = character, i = integer, l = logical, D = date, T = date time, t = time, ? =
94 guess, or _/- to skip a column. For example, read_csv("data.csv", col_types = "i_/-")
95 and an integer, we can use the guess engine to guess the column types. For example,
96 # Your Turn 2
97 Set the 4 column types to integer, character, and unknown (guess)
98 ```{r}
99 csv_data <- read_csv(
100   "a,b,c,d
101    1,2,3,4
102    5,6,7,8",
103   col_types = ""
104 )
105
106 csv_data
107 ...
108
109 ## haven: read and write SAS, SPSS, and Stata files
```

Preview in Window  
✓ Preview in Viewer Pane  
(No Preview)  
✓ Preview Images and Equations  
✓ Show Previews Inline  
Chunk Output Inline  
✓ **Chunk Output in Console**  
Output Options...

readr  
Sample data  
Importing Data  
Your Turn 1  
Tibbles  
Missing values  
Parsing data types  
Your Turn 2  
haven: read and write SA...  
Your Turn 3  
Writing data  
Your Turn 4  
Take Aways

98:7 [1] Chunk 6 [1] R Markdown

Console | Terminal | R Markdown | Jobs

~/Google Drive/Active/r\_teaching\_warehouse/

```
>
> csv_data
> csv_data <- read_csv(
+   "a,b,c,d
+    1,2,3,4
+    5,6,7,8",
+   col_types = ""
+ )
>
> csv_data
# A tibble: 2 x 4
   a     b     c     d
<dbl> <dbl> <dbl> <dbl>
1     1     2     3     4
2     5     6     7     8
>
```

# Project contents

- └─ 01-dplyr\_5verbs
  - | └─ cheatsheet\_dplyr\_5verbs.pdf
  - | └─ diabetes.csv
  - | └─ exercises.Rmd
  - | └─ slides.pdf