Introduction to Tidyverse : : **CHEAT SHEET**

Basics

The **tidyverse** is an opinionated **collection** of R **packages** designed for data science. All packages **share** an underlying philosophy and common APIs.

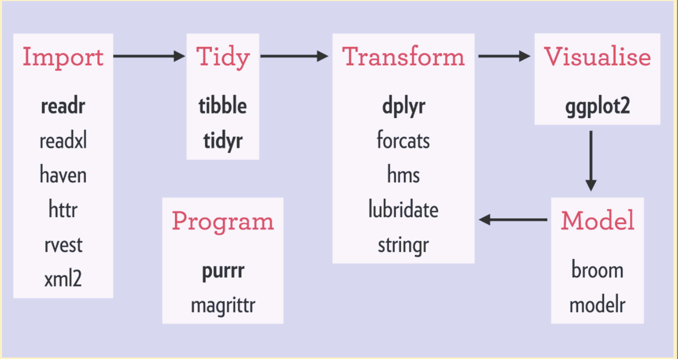
- **dplyr**: data manipulation
- **ggplot2**: creating advanced graphics
- **readr**: importing data
- **tibble**: A tibble, or tbl_df, is a modern reimagining of the data.frame.
- **tidyr**: creating tidy data.
- **purrr**: enhancing R's functional programming.

Why use the tidyverse?

All tidyverse packages and functions serve to accomplish one of two goals:

- Providing **faster**, more **efficient** implementation of base R functions.
- Allow for **cleaner**, **easier** to read syntax.

Workflow of tidyverse:



Importing data with readr

Part of the tidyverse. **readr** provides a **faster** tabular data importing framework compared to base R. Reads and writes more file types than base R and supports reading non-tabular data.

readr functions:

read.csb("file.csv")

read_tsv("file.tsv")

write_excel_csv(df, "file.csv")

read_lines("file.txt")

Manipulating data with tibble

tibble is the tidyverse's rendition of a dataframe. It is part of the **dplyr** package.

We can convert a traditional **dataframe** to a tibble using **as_tibble()**

- **tibble()** never changes the type of the inputs
- **tibble()** never changes the names of variables
- **tibble()** never creates row names

tibble dataframe

	Variable 1	Variable 2
	data type	data type
observation 1		
observation 2		

tibble dataframe example

	Car brand	Model	Year
	<chr>	<chr>	<int>
1	Audi	A4	2015
2	Audi	A8	2015
3	Benz	S200	2016

Piping

Pipe, %>%, one of R's most widely-used functions, aims to make code more readable by reordering the functions so that they appear in the order they are executed.

Without pipe,

head(iris,n=2)

And with pipe

iris %>% head(n=2)

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
```

give the same result.

Transforming data with dplyr

dplyr package allows us to perform data manipulation tasks.

Most data manipulation tasks can be solved using a combination of the following **six functions**:

- **filter**: filters out rows according to some conditions.
- **arrange**: reorders rows according to some conditions.
- **select**: selects a subset of columns.
- **mutate**: adds a new column as a function of existing.
- **summarise**: collapses a data frame to a single row.
- **group_by**: breaks a data frame into groups of rows.

These functions from **dplyr** are designed to be used on a **tibble**, but work on a normal **data frame** as well.

Use iris as an example:

Get "virginica" with Sepal.Length larger than 8:

```
iris %>%
    filter(Species == "virginica", Sepal.Length > 8)
```

Add a column called "Sepal.Area", which values width times length and don't keep Sepal.Length and Sepal.Width:

```
iris %>%
    mutate(Sepal.Area = Sepal.Width *
    Sepal.Length) %>%
    select(-Sepal.Length,-Sepal.Width)
```

Get means of areas each species:

```
iris %>%
    mutate(Sepal.Area = Sepal.Width *
    Sepal.Length) %>%
    group_by(Species) %>%
    summarise(count=n(),
    mean=mean(Sepal.Area))
```

Visualizing data with ggplot2

ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same components: a **data** set, a **coordinate system**, and **geoms**—visual marks that represent data points.

```
ggplot(data = mpg, aes(x = cty, y = hwy))
```

Begins a plot that you finish by adding layers to. Add one geom function per layer.

There is a cheat sheet posted on RStudio, please open the link below for more details of **ggplot2.**

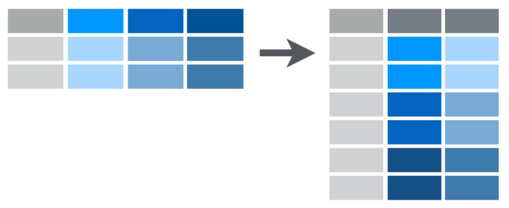
https://raw.githubusercontent.com/rstudio/cheatsheets/main/data-visualization.pdf

Creating tidy data with tidyr

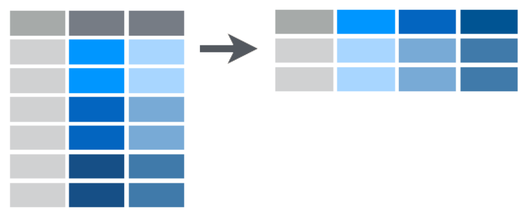
The two main **functions** of **tidyr** are **gather()** and **spread()**. These functions allow **converting** between **long data and wide data** (similar to the reshape package, but better than reshape, and can be used for pipeline %>%).

A data frame where some of the rows **contain information** that is really a **variable name**. This means the columns are a **combination** of variable names as well as some data.

gather() turns wide data to long data like below:



spread() turns long data to wide data like below:



References

Sullivanstatistics. (n.d.). *R basics*. R Basics | Gather. Retrieved March 28, 2022, from http://statseducation.com/Introduction-to-R/modules/tidy%20data/gather/