Data Wrangling in R

Pramod Duvvuri

3/30/2019

Tibble

```
library(tidyverse)
```

```
## -- Attaching packages ----------------------------------------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.1.0       v purrr   0.3.2
## v tibble  2.1.1       v dplyr   0.8.0.1
## v tidyr   0.8.3       v stringr 1.4.0
## v readr   1.3.1       v forcats 0.4.0
```

```
## -- Conflicts -------------------------------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(nycflights13)
```

```
as_tibble(iris) # coerce into a tibble
```

```
## # A tibble: 150 x 5
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##           <dbl>       <dbl>        <dbl>       <dbl> <fct>
##  1          5.1         3.5          1.4         0.2 setosa
##  2          4.9         3            1.4         0.2 setosa
##  3          4.7         3.2          1.3         0.2 setosa
##  4          4.6         3.1          1.5         0.2 setosa
##  5          5           3.6          1.4         0.2 setosa
##  6          5.4         3.9          1.7         0.4 setosa
##  7          4.6         3.4          1.4         0.3 setosa
##  8          5           3.4          1.5         0.2 setosa
##  9          4.4         2.9          1.4         0.2 setosa
## 10          4.9         3.1          1.5         0.1 setosa
## # ... with 140 more rows
```

```
tibble(x = 1:5,
       y = 1,
       z = x ^ 2 + 2 * y)
```

```
## # A tibble: 5 x 3
##       x     y     z
##   <int> <dbl> <dbl>
## 1     1     1     3
## 2     2     1     6
## 3     3     1    11
## 4     4     1    18
## 5     5     1    27
```

```
flights %>% print(n = 10) # width = Inf can be used to print more columns
```

```
## # A tibble: 336,776 x 19
```

```
##     year month   day dep_time sched_dep_time dep_delay arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>
##  1  2013     1     1      517            515         2      830
##  2  2013     1     1      533            529         4      850
##  3  2013     1     1      542            540         2      923
##  4  2013     1     1      544            545        -1     1004
##  5  2013     1     1      554            600        -6      812
##  6  2013     1     1      554            558        -4      740
##  7  2013     1     1      555            600        -5      913
##  8  2013     1     1      557            600        -3      709
##  9  2013     1     1      557            600        -3      838
## 10  2013     1     1      558            600        -2      753
## # ... with 3.368e+05 more rows, and 12 more variables:
## #   sched_arr_time <int>, arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>,
## #   distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
```

```
#flights %>% View()
```

```
df <- tibble(x = runif(5),
             y = rnorm(5))
```

```
df$x
```

```
## [1] 0.73802651 0.54874804 0.77567418 0.49949741 0.01490689
```

```
df[["x"]]
```

```
## [1] 0.73802651 0.54874804 0.77567418 0.49949741 0.01490689
```

```
df[[1]]
```

```
## [1] 0.73802651 0.54874804 0.77567418 0.49949741 0.01490689
```

```
class(mtcars)
```

```
## [1] "data.frame"
```

```
class(df)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

readr

The read_csv() in readr package in faster than the base read.csv() by over (~10x). Also data.table::fread() can read data in at blazing speeds.

1. read_csv()
2. read_csv2()
3. read_tsv()
4. read_fwf()
5. read_log()
6. read_delim()

Parsing Functions

1. parse_logical()
2. parse_integer()

3. parse_date()

tidyr

1. gather()
2. spread()

```
table4a # untidy data
```

```
## # A tibble: 3 x 3
##   country     `1999` `2000`
## * <chr>        <int>  <int>
## 1 Afghanistan    745   2666
## 2 Brazil       37737  80488
## 3 China       212258 213766
```