Introduction to Dates

MSDS 597 Data Wrangling & Husbandry

February 03, 2020

Dates

- All of these are valid ways to write March 1, 2012:
  - 3/1/2012
  - 3/1/12
  - 03/01/12
  - 12/3/1
  - 12/03/01
- Although R has an internal format for dates, you can see that there are potential problems in getting your data into that format.

Using the `lubridate` package (part of the tidyverse)

```r
mdy("3/1/2012")
```

```
## [1] "2012-03-01"
```

```r
mdy("3/1/12")
```

```
## [1] "2012-03-01"
```

```r
mdy("03/01/12")
```

```
## [1] "2012-03-01"
```

```r
ymd("12/3/1")
```

```
## [1] "2012-03-01"
```

```r
ymd("12/03/01")
```

```
## [1] "2012-03-01"
```

```r
ymd("2012-Mar-01")
```

```
## [1] "2012-03-01"
```

```r
mdy("March 1, 2012")
```

```
## [1] "2012-03-01"
```

```r
ymd_hms("2012-03-01 12:23:15")
```

```
## [1] "2012-03-01 12:23:15 UTC"
```

```r
ymd_hms("2012-03-01 12:23:15", tz = "America/New_York")
```

```
## [1] "2012-03-01 12:23:15 EST"
```

The time zone has to be identifiable by your operating system, but there's a list of standard names at (https://en.wikipedia.org/wiki/List_of_tz_database_time_zones)

For me, most of the advantage of the `lubridate` package comes from the ease of parsing dates, but there are many other functions.

```r
example.date <- ymd_hms("2012-03-01 12:23:15", tz = "Americ
round_date(example.date, "hour")
```

```
## [1] "2012-03-01 12:00:00 EST"
```

```r
round_date(example.date, "day")
```

```
## [1] "2012-03-02 EST"
```

```r
round_date(example.date, "month")
```

```
## [1] "2012-03-01 EST"
```

```r
round_date(example.date, "quarter")
```

```
## [1] "2012-04-01 EDT"
```

```r
round_date(example.date, "year")
```

```
## [1] "2012-01-01 EST"
```

Arithmetic with dates

```
example.date
```

```
## [1] "2012-03-01 12:23:15 EST"
```

```
round_date(example.date, "day") + days(3)
```

```
## [1] "2012-03-05 EST"
```

```
round_date(example.date, "day") + months(3)
```

```
## [1] "2012-06-02 EDT"
```

```r
ymd("2012-03-21") - ymd("2012-03-01")
```

```
## Time difference of 20 days
```

```r
as.integer(ymd("2012-03-21") - ymd("2012-03-01"))
```

```
## [1] 20
```

Or change time zones

```r
with_tz(ymd_hms("2012-03-01 12:23:15", tz = "America/New_Yo
```

```
## [1] "2012-03-02 01:23:15 HKT"
```

Or decimal dates

```r
decimal_date(ymd_hms("2012-03-01 12:23:15", tz = "America/N
```

```
## [1] 2012.165
```

There are additional functions to work with timespans (distinguishing among durations, periods, and intervals —see `?lubridate`).

Dates in base R . . .

. . . are harder to parse. You have to specify the format—there's really no reason not to use the `lubridate` package.

```
# Base R
as.Date("3/1/2012", "%m/%d/%Y")
```

```
## [1] "2012-03-01"
```

```
as.Date("3March12", "%d%b%y")
```

```
## [1] "2012-03-03"
```

In class exercise

Using the NYC restaurant inspection data

1. Create a data frame with just the restaurant "SWEET EXPRESSIONS"
2. Simplify what's to come by "fixing" the data frame names with a command like `names(sweet.expressions) <- make.names(names(sweet.expressions))`
3. Change the "INSPECTION DATE" variable from a character string to a Date data type
4. Order the data frame by date.
5. Plot the SCORE against the inspection date.

```
rest <- read_csv('https://data.cityofnewyork.us/api/views/9
```

```
## Parsed with column specification:
## cols(
##   .default = col_character(),
##   CAMIS = col_double(),
##   ZIPCODE = col_double(),
```