Dates & Times

Maria C. Ramos| Cohort 4: R4DS | 07 Apr 2021

mariacramos.com

@mariacramosf

Ice Breaker

Two Truths and One Lie

Outline

- Generalities
- Dates & Times Data Types
- Common Tasks
    - Create data types
    - Pull out individual date components (the month, day, or hour)
    - Round dates
    - Set date components
    - Perform arithmetic operations with dates
    - Work with time zones

Generalities

Package

```
library(lubridate)
```

not part of the tidyverse

Date/Times Data Types

1. date

```
## # A tibble: 1 x 1
##   `my_date <- ymd("2021-04-07")`
##   <date>
## 1 2021-04-07
```

1. date-time

```
## # A tibble: 336,776 x 1
##    time_hour
##    <dttm>
##  1 2013-01-01 05:00:00
##  2 2013-01-01 05:00:00
##  3 2013-01-01 05:00:00
##  4 2013-01-01 05:00:00
##  5 2013-01-01 06:00:00
##  6 2013-01-01 05:00:00
##  7 2013-01-01 06:00:00
##  8 2013-01-01 06:00:00
##  9 2013-01-01 06:00:00
## 10 2013-01-01 06:00:00
## # … with 336,766 more rows
```

Work with the simplest data type you can

Creating date and date times

1. From current date or date time
2. From a string
3. From unquoted numbers
4. From individual date-time components
5. From an existing date, date-time object

Creating date and date times

From current date or date time

Current date

```
today()
```

```
## [1] "2021-04-07"
```

Current date-time

```
now()
```

```
## [1] "2021-04-07 17:55:00 -05"
```

Creating date and date times

From strings

Approach 1: parsing (data import)

Approach 2: using lubridate helpers

- Helpers: y, m, d to represent year, month, day, etc.
- Identify the order in which the components appear in the data
- Specify the order of the components in your data to parse

```
ymd("2017-01-31")
```

```
## [1] "2017-01-31"
```

```
myd("01-2017-31")
```

```
## [1] "2017-01-31"
```

Creating date and date times

From strings (examples)

```
mdy("January 31st, 2017")
```

```
## [1] "2017-01-31"
```

```
dmy("31-Jan-2017")
```

```
## [1] "2017-01-31"
```

```
ymd_hms("2017-01-31 20:11:59")
```

```
## [1] "2017-01-31 20:11:59 UTC"
```

```
mdy_hm("01/31/2017 08:01")
```

```
## [1] "2017-01-31 08:01:00 UTC"
```

Exercise

Which of helpers will correctly parse the date-time objects?

"30/06/2020 06:01:01"

```
myd("30/06/2020 06:01")
mdy_hms("30/06/2020 06:01:01")
dmy_hms("30/06/2020 06:01:01")
none of the above
```

"April 7, 2021"

```
myd_hms("April 7, 2021")
mdy_hms("April 7, 2021")
ymd("April 7, 2021")
none of the above
```

Creating date and date times

From unquoted number

```
ymd(20170131)
```

```
## [1] "2017-01-31"
```

Adding the timezone

```
ymd(20170131, tz = "UTC")
```

```
## [1] "2017-01-31 UTC"
```

Creating date and date times

From individual components

use *make_date()* and *make_datetime()*

```
## # A tibble: 336,776 x 6
##    flight  year month   day  hour minute
##     <int> <int> <int> <int> <dbl>  <dbl>
##  1   1545  2013     1     1     5     15
##  2   1714  2013     1     1     5     29
##  3   1141  2013     1     1     5     40
##  4    725  2013     1     1     5     45
##  5    461  2013     1     1     6      0
##  6   1696  2013     1     1     5     58
##  7    507  2013     1     1     6      0
##  8   5708  2013     1     1     6      0
##  9     79  2013     1     1     6      0
## 10    301  2013     1     1     6      0
## # … with 336,766 more rows
```

Creating date and date times

From individual components

use *make_date()* and *make_datetime()*

```
flights %>%
   select(flight, year, month, day, hour, minute) %>%
   mutate(departure = make_date(year, month, day))%>%
   print(n=5)
```

```
## # A tibble: 336,776 x 7
##   flight  year month   day  hour minute departure
##    <int> <int> <int> <int> <dbl>  <dbl> <date>
## 1   1545  2013     1     1     5     15 2013-01-01
## 2   1714  2013     1     1     5     29 2013-01-01
## 3   1141  2013     1     1     5     40 2013-01-01
## 4    725  2013     1     1     5     45 2013-01-01
## 5    461  2013     1     1     6      0 2013-01-01
## # … with 336,771 more rows
```

Creating date and date times

From an existing date, date-time object

To switch between date and date-time objects

```
as_datetime(today())
```

```
## [1] "2021-04-07 UTC"
```

```
as_date(now())
```

```
## [1] "2021-04-07"
```

Pulling out components

accessor function
year()
month()
mday()
yday()
wday()
hour()
minute()
second()

Pulling out components

```
datetime <-
   ymd_hms("2021-04-07 12:34:56"
```

```
year(datetime)
```

```
## [1] 2021
```

```
month(datetime)
```

```
## [1] 4
```

```
mday(datetime)
```

```
## [1] 7
```

```
yday(datetime)
```

```
## [1] 97
```

```
wday(datetime)
```

```
## [1] 4
```

Pulling out components

```
datetime <-
   ymd_hms("2021-04-07 12:34:56"
```

```
month(datetime,
      label = TRUE)
```

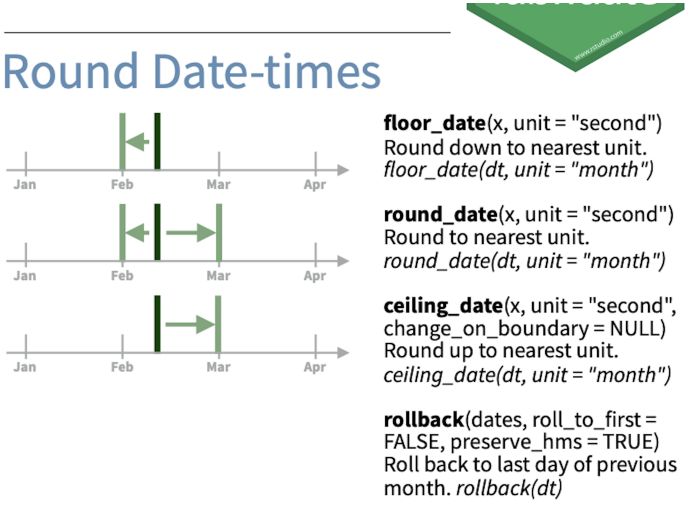
```
## [1] Apr
## 12 Levels: Jan < Feb < Mar < Apr < Ma
```

```
wday(datetime,
     label = TRUE,
     week_start = 1)
```

```
## [1] Wed
## Levels: Mon < Tue < Wed < Thu < Fri <
```

Rounding Date-times

Rounding Date-times




```
x <- now()
floor_date(x, "month")
```

```
## [1] "2021-04-01 -05"
```

```
round_date(x, "month")
```

```
## [1] "2021-04-01 -05"
```

```
ceiling_date(x,"month")
```

```
## [1] "2021-05-01 -05"
```

```
rollback(x)
```

```
## [1] "2021-03-31 17:55:00 -05"
```

Exercises

What should each of these rounding functions output?

```
my_date <-  mdy_hms("01-26-2021 16:35:00")
       floor_date(my_date, "hour")
       round_date(my_date, "hour")
      ceiling_date(my_date,"hour")
           rollback(my_date)
```

Solutions

```
my_date <-  mdy_hms("01-26-2021 16:35:00")
floor_date(my_date, "hour")
```

```
## [1] "2021-01-26 16:00:00 UTC"
```

```
round_date(my_date, "hour")
```

```
## [1] "2021-01-26 17:00:00 UTC"
```

```
ceiling_date(my_date,"hour")
```

```
## [1] "2021-01-26 17:00:00 UTC"
```

```
rollback(my_date)
```

```
## [1] "2020-12-31 16:35:00 UTC"
```

Setting components

Using accessor functions

setting the year

```
x
```

```
## [1] "2021-04-07 17:55:00 -05"
```

```
year(x) <- 2020
x
```

```
## [1] "2020-04-07 17:55:00 -05"
```

setting the month

```
month(x) <- month(x)+1
x
```

```
## [1] "2020-05-07 17:55:00 -05"
```

Setting components

Using update() function

updating year

```
x
```

```
## [1] "2020-05-07 17:55:00 -05"
```

```
update(x, year = 2019)
```

```
## [1] "2019-05-07 17:55:00 -05"
```

```
x
```

```
## [1] "2020-05-07 17:55:00 -05"
```

Setting components

Using update() function

using rollover

```
ymd("2021-04-07") %>%
   update(mday = 31)
```

```
## [1] "2021-05-01"
```

Next Week

Time Spans

Thanks!

Slides created via the R package **xaringan**.