Ngày tháng & thời gian với lubridate : : CHEAT SHEET

Date-times

2017-11-28 12:00:00
date-time là kiểu dữ liệu thời gian, được tính bằng số giây kể từ ngày 1970-01-01 00:00:00 UTC

*dt <- **as_datetime**(1511870400)*
"2017-11-28 12:00:00 UTC"

2017-11-28
date là một kiểu dữ liệu thời gian, tính bằng số ngày kể từ ngày 1970-01-01

*d <- **as_date**(17498)*
"2017-11-28"

12:00:00
hms là kiểu dữ liệu thời gian, tính bằng số giây kể từ thời điểm 00:00:00

*t <- hms::**as.hms**(85)*
00:01:25

Biến đổi dữ liệu sang định dạng date-time

1. Trong cấu trúc dữ liệu, xác định thứ tự của các biến năm (**y**), tháng (**m**), ngày (**d**), giờ (**h**), phút (**m**) và giây (**s**).
2. Sử dụng các hàm dưới đây, tên mỗi hàm thể hiện thứ tự của các biến thời gian. Mỗi hàm chấp nhận nhiều kiểu định dạng dữ liệu đầu vào.

2017-11-28T14:02:00 — **ymd_hms**(), **ymd_hm**(), **ymd_h**(). *ymd_hms("2017-11-28T14:02:00")*

2017-22-12 10:00:00 — **ydm_hms**(), **ydm_hm**(), **ydm_h**(). *ydm_hms("2017-22-12 10:00:00")*

11/28/2017 1:02:03 — **mdy_hms**(), **mdy_hm**(), **mdy_h**(). *mdy_hms("11/28/2017 1:02:03")*

1 Jan 2017 23:59:59 — **dmy_hms**(), **dmy_hm**(), **dmy_h**(). *dmy_hms("1 Jan 2017 23:59:59")*

20170131 — **ymd**(), **ydm**(). *ymd(20170131)*

July 4th, 2000 — **mdy**(), **myd**(). *mdy("July 4th, 2000")*

4th of July '99 — **dmy**(), **dym**(). *dmy("4th of July '99")*

2001: Q3 — **yq**() Q thể hiện quý. *yq("2001: Q3")*

2:01 — hms::**hms**() & lubridate::**hms**(), **hm**() và **ms**(), trả ra kết quả khoảng thời gian.* *hms::hms(sec = 0, min= 1, hours = 2)*

2017.5 — **date_decimal**(decimal, tz = "UTC") *date_decimal(2017.5)*

now(tzone = "") Thời gian trong múi giờ (mặc định theo múi giờ của máy tính). *now()*

today(tzone = "") Ngày hiện tại theo múi giờ (mặc định theo múi giờ của máy tính). *today()*

fast_strptime() Biến đổi nhanh sang định dạng dữ liệu.
fast_strptime('9/1/01', '%y/%m/%d')

parse_date_time() Biến đổi sang định dạng dữ liệu.
parse_date_time("9/1/01", "ymd")

Trích xuất & biến đổi dữ liệu

Sử dụng các hàm thời gian để trích xuất dữ liệu.
Gán dữ liệu với một giá trị để thay đổi dữ liệu khi cần thiết

d ## "2017-11-28"
day(d) ## 28
day(d) <- 1
d ## "2017-11-01"

2018-01-31 11:59:59 — **date**(x) Ngày. *date(dt)*

2018-01-31 11:59:59 — **year**(x) Năm. *year(dt)*
isoyear(x) Năm - ISO 8601.

2018-01-31 11:59:59 — **month**(x, label, abbr) Tháng. *month(dt)*
day(x) Ngày trong tháng. *day(dt)*
wday(x,label,abbr) Ngày trong tuần.
qday(x) Ngày trong quý.

2018-01-31 11:59:59 — **hour**(x) Giờ. *hour(dt)*

2018-01-31 11:59:59 — **minute**(x) Phút. *minute(dt)*

2018-01-31 11:59:59 — **second**(x) Giây. *second(dt)*

week(x) Tuần trong năm. *week(dt)*

isoweek() Tuần - ISO 8601.
epiweek() Tuần

quarter(x, with_year = FALSE) Quý. *quarter(dt)*

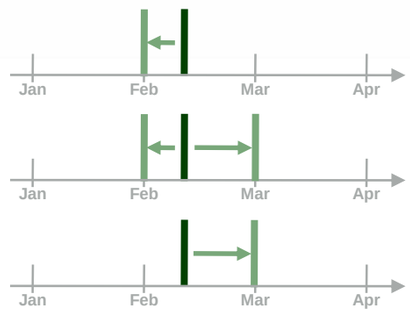
semester(x, with_year = FALSE) Nửa năm. *semester(dt)*
am(x) x là buổi sáng? *am(dt)*
pm(x) x là buổi chiều? *pm(dt)*

dst(x) Là ngày đổi múi giờ? *dst(d)*

leap_year(x) x là năm nhuận? *leap_year(d)*

update(object, ..., simple = FALSE) *update(dt, mday = 2, hour = 1)*

Làm tròn thời gian



floor_date(x, unit = "second") Làm tròn xuống theo đơn vị cho trước
floor_date(dt, unit = "month")

round_date(x, unit = "second") Làm tròn lên theo đơn vị cho trước
round_date(dt, unit = "month")

ceiling_date(x, unit = "second", change_on_boundary = NULL) Làm tròn đến đơn vị gần nhất.
ceiling_date(dt, unit = "month")

rollback(dates, roll_to_first = FALSE, preserve_hms = TRUE) Quay ngược lại ngày cuối cùng của tháng trước
rollback(dt)

Tạo mẫu thời gian

stamp() Tạo một mẫu thời gian từ ví dụ và trả ra kết quả là một hàm áp dụng mẫu vừa tạo với kiểu dữ liệu thời gian. Tương tự với **stamp_date**() & **stamp_time**().

1. Tạo template mới
   sf <- stamp("Created Sunday, Jan 17, 1999 3:34")
2. Sử dụng template mới vừa tạo
   sf(ymd("2010-04-05"))
   ## [1] "Created Monday, Apr 05, 2010 00:00"

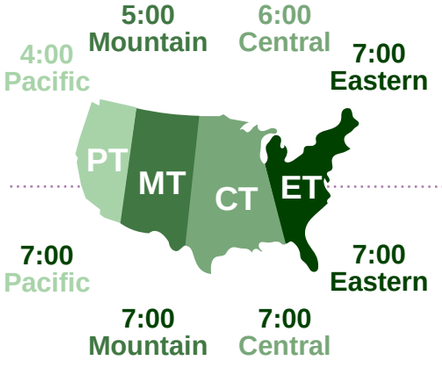
Tip: sử dụng date > 12

Múi giờ

R nhận dạng ~600 múi giờ khác nhau. Mỗi múi giờ sẽ có chứa các thông tin chi tiết về múi giờ, thời gian đổi lịch mùa đông, mùa xuân và lịch sử thay đổi của lịch cho khu vực. R gán mỗi múi giờ với một vector.

Sử dụng múi giờ **UTC** để tránh ảnh hưởng của việc đổi lịch thời gian mùa đông/mùa hè.

OlsonNames()Trả ra danh sách các múi giờ. *OlsonNames()*



with_tz(time, tzone = "") Thời gian tại múi giờ khác của biến time.
with_tz(dt, "US/Pacific")

force_tz(time, tzone = "") Trả ra thời gian có giá trị như đầu vào time nhưng ở múi giờ khác.
force_tz(dt, "US/Pacific")

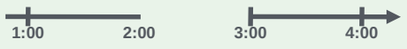
Translator: ranalytics.vn
Xem thêm lubridate.tidyverse.org • lubridate 1.6.0 • Updated: 2017-12

Tính toán dữ liệu thời gian — lubridate cung cấp 3 định dạng dữ liệu khác nhau để thực hiện phép tính với thời gian

Khi tính toán, các mốc thời gian (timeline) có thể không đồng nhất với nhau do sự khác biệt về múi giờ. Xem các ví dụ dưới đây:

Ngày bình thường
nor <- ymd_hms("2018-01-01 01:30:00",tz="US/Eastern")

Ngày bắt đầu đổi múi giờ sang hè (nhanh hơn 1h so với bình thường
gap <- ymd_hms("2018-03-11 1:30:00",tz="US/Eastern")

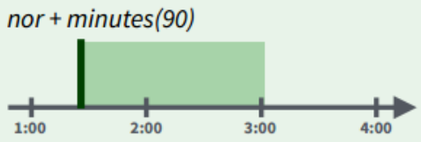


Ngày đổi múi giờ sang mùa đông (chậm hơn 1h)
lap <- ymd_hms("2018-11-04 00:30:00",tz="US/Eastern")

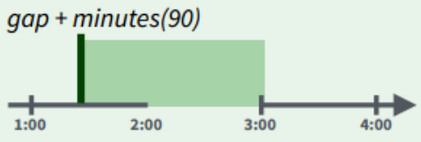
Năm nhuận và giây nhuận
leap <- ymd("2019-03-01")

Periods theo dõi sự thay đổi về mặt thời gian, bỏ qua các trường hợp bất quy tắc

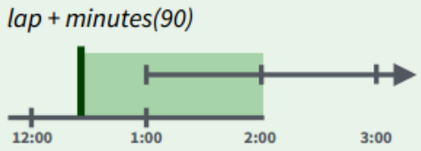
nor + minutes(90)



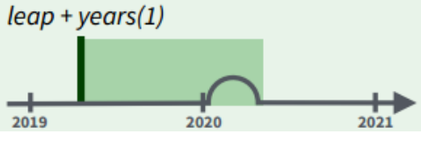
gap + minutes(90)



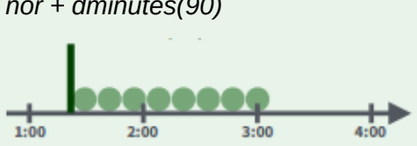
lap + minutes(90)



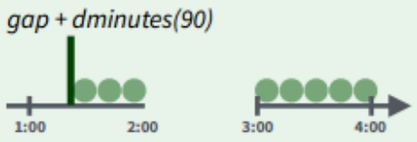
leap + years(1)



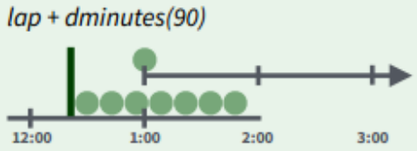
Durations theo dõi sự thay đổi của dòng thời gian thực tế, có tính đến các trường hợp bất quy tắc.
nor + dminutes(90)



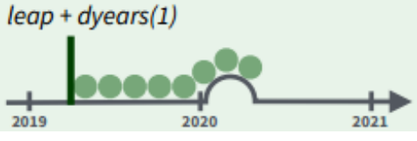
gap + dminutes(90)



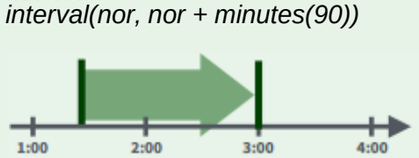
lap + dminutes(90)



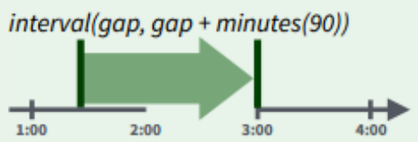
leap + dyears(1)



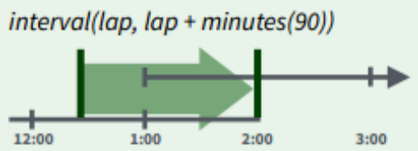
Intervals mô tả một khoảng thời gian cụ thể, có thời điểm bắt đầu và kết thúc.
interval(nor, nor + minutes(90))



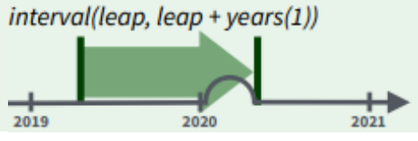
interval(gap, gap + minutes(90))



interval(lap, lap + minutes(90))



interval(leap, leap + years(1))



Không phải tất cả các năm đều có 365 ngày do có **ngày nhuận**. Không phải lúc nào một phút cũng có 60s do có **giây nhuận**.

Ta có thể tạo ngày trong tháng tưởng tượng bằng các thêm **months**, VD:31 tháng 2
jan31 <- ymd(20180131)
jan31 + months(1)

NA

%m+% & **%m-%** sẽ lùi ngày tưởng tượng về ngày cuối cùng của tháng trước đó.
jan31 %m+% months(1)

"2018-02-28"

add_with_rollback(e1, e2, roll_to_first = TRUE) sẽ chuyển ngày tưởng tượng vào ngày đầu tiên của tháng mới.
add_with_rollback(jan31, months(1), roll_to_first = TRUE)

"2018-03-01"

PERIODS

Sử dụng dữ liệu định dạng periods để tính toán/mô hình hóa các sự kiện xảy ra tại các điểm thời gian.

Tạo biến khoảng thời gian với các đơn vị thời gian từ lớn đến nhỏ.

p <- months(3) + days(12)
p
"3m 12d 0H 0M 0S"

Số tháng | Số ngày | etc.

years(x = 1) x năm.
months(x) x tháng.
weeks(x = 1) x tuần.
days(x = 1) x ngày.
hours(x = 1) x giờ.
minutes(x = 1) x phút.
seconds(x = 1) x giây.
milliseconds(x = 1) x milli giây.
microseconds(x = 1) x micro giây
nanoseconds(x = 1) x nano giây.
picoseconds(x = 1) x pico giây.

period(num = NULL, units = "second", ...) Tự động tính toán khoảng thời gian. *period(5, unit = "years")*

as.period(x, unit) Biến đổi dữ liệu sang dạng period. Xem thêm **is.period**(). *as.period(i)*

period_to_seconds(x) Biến đổi dữ liệu khoảng thời gian thành giây. Xem thêm **seconds_to_period**(). *period_to_seconds(p)*

DURATIONS

Dữ liệu định dạng duration dùng để tính toán/mô hình hóa các quá trình vật lý, như tuổi thọ của pin. Định dạng dữ liệu durations được lưu dưới dạng giây. **Difftimes** là cấu trúc dữ liệu của duration được cài đặt mặc định trong R.

Định dạng duration có tiền tố ***d*** *trước các câu lệnh*, ví dụ.

dd <- ddays(14)
dd
"1209600s (~2 tuần)"

Thời gian tính bằng giây | Thời gian quy đổi ra các định dạng thông thường khác

dyears(x = 1) 31536000x giây.
dweeks(x = 1) 604800x giây.
ddays(x = 1) 86400x giây.
dhours(x = 1) 3600x giây.
dminutes(x = 1) 60x giây.
dseconds(x = 1) x giây.
dmilliseconds(x = 1) x 10^{-3} giây.
dmicroseconds(x = 1) x 10^{-6} giây.
dnanoseconds(x = 1) x 10^{-9} giây.
dpicoseconds(x = 1) x 10^{-12} giây.

duration(num = NULL, units = "second", ...) Tính toán nhanh sang định dạng duration. *duration(5, unit = "years")*

as.duration(x, …) Biến đổi sang định dạng duration. Also **is.duration**(), **is.difftime**(). *as.duration(i)*

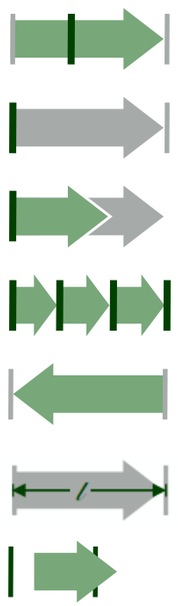
make_difftime(x) Tạo sự khác biệt về thời gian với số lượng cho trước các đơn vị. *make_difftime(99999)*

INTERVALS

Dữ liệu định dạng interval cho phép ta xác định độ dài của khoảng thời gian.

Tạo interval với hàm **interval**() hoặc **%--%**.
*i <- **interval**(ymd("2017-01-01"), d)* *## 2017-01-01 UTC--2017-11-28 UTC*
*j <- d **%--%** ymd("2017-12-31")* *## 2017-11-28 UTC--2017-12-31 UTC*

Start Date | End Date



a **%within%** b a có nằm trong b không?
now() %within% i

int_start(int) Thời điểm bắt đầu của interval. Xem thêm**int_end**(). *int_start(i) <- now(); int_start(i)*

int_aligns(int1, int2) Kiểm tra hai interval có chồng lấn không? Xem thêm **int_overlaps**(). *int_aligns(i, j)*

int_diff(times) Tạo interval trong một véc-tơ thời gian. *v <-c(dt, dt + 100, dt + 1000); int_diff(v)*

int_flip(int) Đảo chiều của một interval. Xem **int_standardize**(). *int_flip(i)*

int_length(int) Độ dài theo giây. *int_length(i)*
int_shift(int, by) Dịch chuyển interval trong một khoảng thời gian. *int_shift(i, days(-1))*

as.interval(x, start, …) Biến đổi x thành interval với thời gian bắt đầu cho trước, xem **is.interval**(). *as.interval(days(1), start = now())*

Translator: ranalytics.vn