使用背景：在我们处理一些时间序列数据时，经常会碰到各种时间数据，比如“2017-03-03”。很多时候我们需要提取出其中的年、月、日甚至是小时、分、秒，从而可以方便的进行比较、筛选等操作。如果我们自己去实现上述功能，可能会写一个字符串的提取函数，来确定相应的时间单位值。但是，由于时间数据格式多样，总会碰到一些问题。还好lubridate这个包实现了各种功能，功能简单但方便快捷。

目录 | Contents

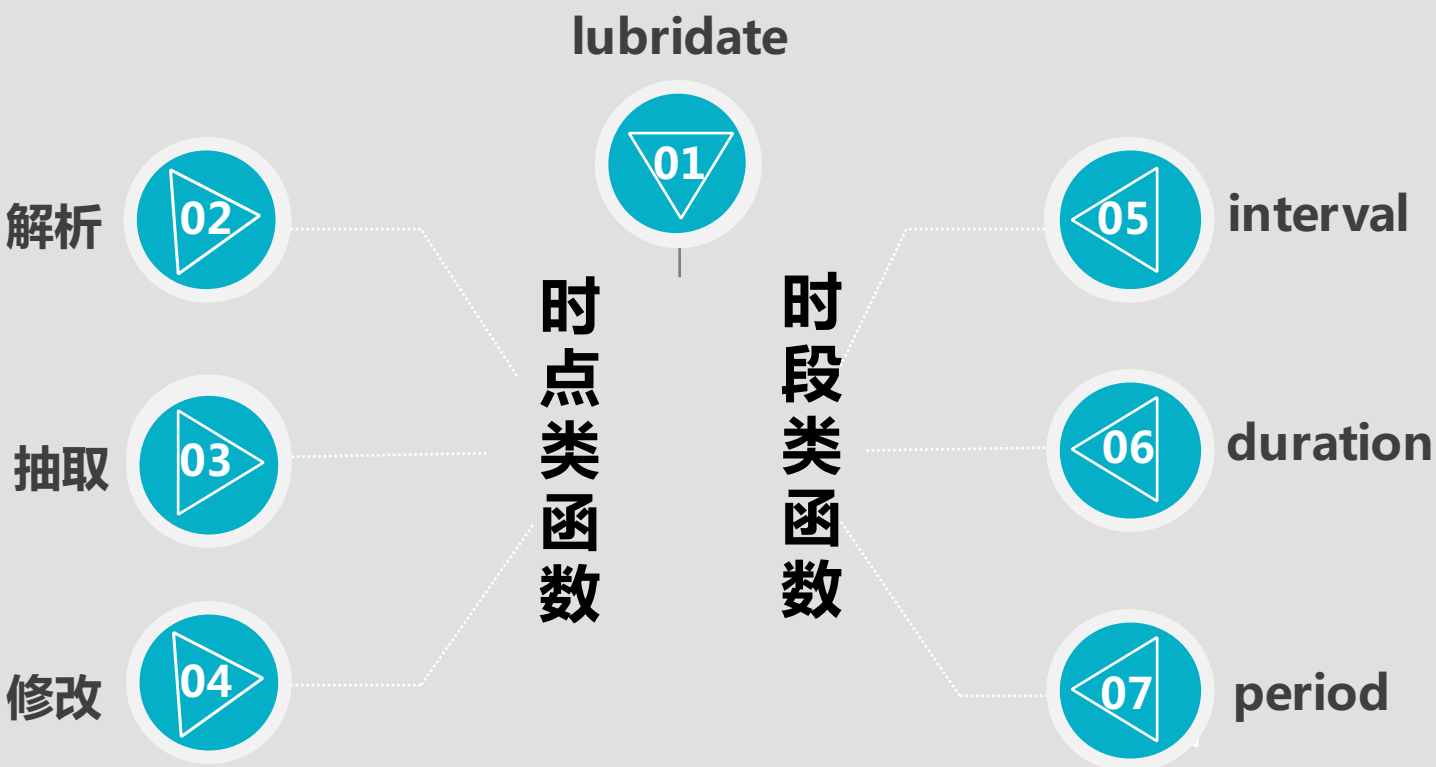
概念分类

示例分析

习题案例

Lubridate包主要函数：
处理时点数据函数和处理时段数据函数

1


lubridate
01
时点类函数
时段类函数
解析
02
抽取
03
修改
04
05
interval
06
duration
07
period

```
# 从字符型数据解析时间，会自动识别各种分隔符
> x <- ymd('2017-04-26')
# 观察x日期是一年中的第几天
> yday(x)
# 修改x日期中的月份为5月
> month(x) <- 5
```

interval：最简单的时段对象，它由两个时点数据构成。
duration：去除了时间两端的信息，纯粹以秒为单位计算时段的长度，不考虑闰年和闰秒，它同时也兼容基本包中的difftime类型对象。
period：以较长的时钟周期来计算时段长度，它考虑了闰年和闰秒，适用于长期的时间计算。以2012年为例，duration计算的一年是标准不变的365天，而period计算的一年就会变成366天。
有了时点和时段数据，就可以进行各种计算了。

为了处理时区信息，lubridate包提供了三个函数：
tz：提取时间数据的时区
with_tz：将时间数据转换为另一个时区的同一时间
force_tz：将时间数据的时区强制转换为另一个时区

```
# 输入欧洲杯决赛在乌克兰的开场时间，再转为北京时间
eurotime <- ymd_hms('2012-07-01 21:45:00',tz='EET')
with_tz(eurotime,tzone='asia/shanghai')
[1] "2012-07-02 02:45:00 CST"
```

示例分析

2

返回时间值

- lubridate函数的方便之处在于无论年月日之间以什么间隔符分隔，它总能找到正确的值且返回的是数字值
- ymd，mdy，dmy分别表示了三种常见的年月日排列方式，可以把不同的日期数据转化为标准的日期数据。

```
> year("2016-10-24")
[1] 2016
> year("2016/10/24")
[1] 2016
> month("2016/10/24")
[1] 10
> day("2016/10/24")
[1] 24
```

```
> ymd("20110604")
[1] "2011-06-04"
> mdy("06-04-2011")
[1] "2011-06-04"
> dmy("04/06/2011")
[1] "2011-06-04"
```

时间数据运算

- 我们可以看到有两个函数：minutes()，dminutes()，minutes(2)函数表示的2个整分钟的概念，而dminutes()则是具体120秒的概念。
- leap_year()函数可以判断是否是闰年，dyears(1)表示的365天，而years(1)则是一个整年的概念。

```
> minutes(2) ## period
[1] "2M 0S"
> dminutes(2) ## duration
[1] "120s (~2 minutes)"
```

```
> leap_year(2011) ## regular year
[1] FALSE
> ymd(20110101) + dyears(1)
[1] "2012-01-01"
> ymd(20110101) + years(1)
[1] "2012-01-01"
> leap_year(2012) ## leap year
[1] TRUE
> ymd(20120101) + dyears(1)
[1] "2012-12-31"
ymd(20120101) + years(1)
> [1] "2013-01-01"
```

时间区间

- lubridate还允许我们定义一个时间区间
- 有了时间区间的定义，我们还可以判断一个时间区间是否在另一个时间区间里面，用"%within%"操作符。

```
> arrive<-"2011-08-10 14:00:00"
> leave<-"2011-08-10 14:00:05"
> int<-interval(arrive,leave)
[1] 2011-08-10 14:00:00 UTC--2011-08-10 14:00:05 UTC
```

```
> as.period(int1)
[1] "10M 9S"
> int1 / dminutes(1)
[1] 10.15
```

```
> arrive1<-"2011-08-10 13:50:00"
> leave1<-"2011-08-10 14:00:09"
> int1<-interval(arrive1,leave1)
> int1 %within% int
[1] FALSE
> int %within% int1
[1] TRUE
```

03

习题案例

1 | 习题

➢#奥克兰9点（2017年4月26日）电话会议，求芝加哥几点开。

#芝加哥如果误认为9点开，那么奥克兰接到电话的时间是多少。

- 在金融市场中谣传着一种日历效应。即在一周的第一天或者一年的第一个月份，股票会出现不错的上涨。让我们用热图来观察一下。首先是获取上证指数数据，然后根据不同的月份和星期数，将收益率汇集到不同的组中。将该组收益率的中位数映射到图形颜色上去。从图中判断是否存在明显周一效应或是一月效应。

谢谢观看

解答1

```
#奥克兰9点（2017年4月26日）电话会议，求芝加哥几点开
meeting <- ymd_hms("2017-04-26 09:00:00", tz = "Pacific/Auckland")
with_tz(meeting, "America/Chicago")
#芝加哥如果误认为9点开，那么奥克兰接到电话的时间是多少
mistake <- force_tz(meeting, "America/Chicago")
with_tz(mistake, "Pacific/Auckland")
```

解答2

```
library(quantmod)
library(ggplot2)
library(lubridate)
# 读取上证指数历史数据
getSymbols('^SSEC',src='yahoo',from  = '1997-01-01')
time <- ymd(as.character(index(SSEC)))
open <- as.numeric(Op(SSEC))
high <- as.numeric(Hi(SSEC))
low <- as.numeric(Lo(SSEC))
close <- as.numeric(Cl(SSEC))
volume <- as.numeric(Vo(SSEC))
# 根据收盘和开盘计算当日收益率
profit <- (close-open)/open
# 提取时间数据中的周数和月份
wday <- wday(time)-1
mday <-month(time)
data <- data.frame(time,wday,mday,profit)
p <- ggplot(data,aes(factor(mday),factor(wday),z=profit))
# 收益率的热图，图中颜色越浅，表示汇集到这个组中的收益率中位数越高
p +stat_summary2d(fun=function(x)  median(x))+
  opts(legend.position  = "top")+ labs(x='月份',y='星期')
```