# Introducing time based queries

# ISO 8601:2004

- International standard for date and time
- Left to right from most to least significant digit
- "YYYY-MM-DDTHH:MM:SS" format
    - `"2014"` **OK**
    - ~~`"02"`~~

# `xts` support of ISO 8601:2004

- One and two sided intervals `"2004"` & `"2001/2015"`
- Truncated representation `"201402/03"`
- Time support `"2014-02-22 08:30:00"`
- Repeating intervals `"T08:00/T09:00"`

# One & two sided intervals

```
> # Load fund data
> data(edhec, package = "PerformanceAnalytics")

> head(edhec["2007-01", 1])
           Convertible Arbitrage
2007-01-31                 0.013
```

```
> head(edhec["2007-01/2007-03", 1])
          Convertible Arbitrage
2007-01-31                0.0130
2007-02-28                0.0117
2007-03-31                0.0060
```

# Truncated dates

```
> # January 2007 to March

> head(edhec["200701/03", 1])
           Convertible Arbitrage
2007-01-31                0.0130
2007-02-28                0.0117
2007-03-31                0.0060
```

# Time support

```
> # YYYYMMDDTHHMM format
> iday["20160808T2213"]
                     [,1]
2016-08-08 22:13:02 8.56
2016-08-08 22:13:25 7.71
2016-08-08 22:13:41 8.40
2016-08-08 22:13:55 7.94
2016-08-08 22:13:59 9.29
```

# Repeating intraday intervals

```
> iday["T05:30/T06:30"]
                     [,1]
2016-08-12 05:30:31 12.47
2016-08-16 06:07:54 10.49
2016-08-16 06:10:03  8.94
2016-08-17 06:18:08  9.29
```

MANIPULATING TIME SERIES DATA IN R WITH XTS & ZOO

# Let’s practice!

MANIPULATING TIME SERIES DATA IN R WITH XTS & ZOO

# Alternative extraction techniques

# Row selection with time

- Integer indexing

```
> x[c(1, 2, 3), ]
```

- Logical vectors

```
> x[index(x) > "2016-08-20"]
```

- Date objects (`Date`, `POSIXct`, etc.)

```
dates <- as.POSIXct(c("2016-06-25", "2016-06-27"))
x[dates]
```

# Modifying time series

- Same flexibility as subsetting
  - ISO 8601, integers, logicals, and date objects
- `which.i = TRUE` creates an integer vector corresponding to times

```
> index <- x["2007-06-26/2007-06-28", which.i = TRUE]
> index
[1] 2 3 4
```

# Key behaviors

- All subsets preserve matrix (`drop = FALSE`)
- Order is preserved
- Binary search and `memcpy` are faster than base R!
- `index` and `xts` attributes are preserved

# Let’s practice!

MANIPULATING TIME SERIES DATA IN R WITH XTS & ZOO

# Methods to find periods in your data

# Finding times of interest

- R uses `head()` & `tail()` to look at the start or end of a series
- `xts` implements 2 similar functions with respect to time
  - Uses a flexible notion of time
  - i.e. “last 3 days” or “first 6 weeks”
- These are the `first()` and `last()` functions

# first() and last()

```
> first(edhec[, "Funds of Funds"], "4 months")
           Funds of Funds
1997-01-31         0.0317
1997-02-28         0.0106
1997-03-31        -0.0077
1997-04-30         0.0009
```

```
> last(edhec[, "Funds of Funds"], "1 year")
           Funds of Funds
2009-01-31         0.0060
2009-02-28        -0.0037
2009-03-31         0.0008
2009-04-30         0.0092
2009-05-31         0.0312
2009-06-30         0.0024
2009-07-31         0.0153
2009-08-31         0.0113
```

# first() and last()

- n can also be an integer
- n = 10, n = 2, etc.
    - n = "6 hours"
    - n = "-6 months"

```
first(x, n = 1, keep = FALSE)

last(x, n = 1, keep = FALSE)
```

# Combine function calls

- `first()` and `last()` can be nested for internal intervals
    - Used to find start or end periods within others

```
> first(last(edhec[, "Merger Arbitrage"], "2 years"), "5 months")
           Merger Arbitrage
2008-01-31          -0.0126
2008-02-29           0.0060
2008-03-31          -0.0045
2008-04-30           0.0149
2008-05-31           0.0136
```

MANIPULATING TIME SERIES DATA IN R WITH XTS & ZOO

# Let’s practice!

# Math operations using `xts`

# Key features

- `xts` is naturally a matrix
- Math operations are on the ***intersection*** of times
  - Only these intersections will be used
- Sometimes it is necessary to drop the `xts` class
  - argument `drop = TRUE`, `coredata()`, or `as.numeric()`
- Special handling required for ***union*** of dates

# Out of the box ops (+, −, *, /)

```
> x
           x
2016-08-09 1
2016-08-10 1
2016-08-11 1

> y
           y
2016-08-09 2
2016-08-10 2
2016-08-12 2

> x + y       # Intersection of dates
           x
2016-08-09 3
2016-08-10 3
```

# Operations on the union

- It may be necessary to use all observations
- Covered in detail next chapter

```
> x_union <- merge(x, index(y), fill = 0)

> y_union <- merge(y, index(x), fill = 0)

> x_union + y_union
           x
2016-08-09 3
2016-08-10 3
2016-08-11 1
2016-08-12 2
```

MANIPULATING TIME SERIES DATA IN R WITH XTS & ZOO

# Let’s practice!