RQAD – Walkthrough

This document shows a very basic example using the RQAD package to gather data from the QAD database .

This assumes that the RQAD package has been downloaded and installed, but that no further configuration has yet been performed.

Connecting to the database

Before RQAD can be used, it must be given the connection information to your locally available installation of QA Direct. This is accomplished through the **set.qad.connection** function, which takes as its argument a function that returns a valid QAD database connection.

My local database is a SQL Server installation, and as such I can simply pass the **example.connection** function with my credentials:

```
set.qad.connection(example.connection, "qad.internal.thomsonreuters.com ", "ian",
"password1", "database=qai")
```

All future data-request calls will now be directed using these credentials.

Specifying identifiers

QAD uses as its primary identifier a *seccode*, but multiple other identifiers exist within QAD, such as CIK codes and RICs. In this experiment, the equities I wish to investigate are a number of technology companies denoted by their RIC, so I must first convert these to seccodes with the **get.seccodes.for.rics** function. Similar functions exist to translate between other identifiers.

```
rics <- c("MSFT.OQ", "GOOG.OQ", "AAPL.OQ", "IBM.N", "BIDU.OQ", "HPQ.N", "SNE.N",
"ADBE.OQ")
seccodes <- get.seccodes.for.rics(rics)$seccode
```

Retrieving data

RQAD contains a number of convenience functions for retrieving commonly used data. Other fields are available by name through, e.g., the **get.info.from.rkd** function for RKD fields.

In this case we want to get earnings surprise data, and adjusted daily close prices for the equities above, in a given time period for our experiment, using the **get.earnings.surprise** and **get.adj.daily.close** functions.

```
dates <- seq(as.Date("2008-01-01"), by="1 day", to=as.Date("2011-01-01"))
earnings.surprise <- get.earnings.surprise(dates, seccodes)
close.prices <- get.adj.daily.close(dates,seccodes,per.seccode=1)
```

Note that the default behavior for functions such as **get.adj.daily.close** above is to use **per.seccode=1**. This submits a query to the database which explicitly specifies which identifiers to query, and would not work for a large (around 500-1000, depending on your DB configuration) number of identifiers. The alternative setting of **per.seccode=0** would return the entire table and filter it in R, which is inefficient for a small number of identifiers. Here, then, we set this parameter to 1.

Preparing data

A quick summary of our earnings surprise data shows that it contains mostly NA entries, given that earnings (and therefore earnings surprise) are only reported a few times in any given year. Thus, we will filter to just those dates on which we have earnings.

```
# Get the percentage of NAs, per identifier, in our earnings.surprise data
apply(earnings.surprise, 2, FUN=function(x) sum(is.na(x))/length(x))
```

```
     2099     46692     30902     33402     70436     36799      6027
0.9890611 0.9890611 0.9890611 0.9890611 0.9890611 0.9890611 0.9890611
```

```
earnings.surprise <- earnings.surprise[apply(!is.na(earnings.surprise), 1, any), ]
```

Finally, where earnings are not reported on a trading day, we need the date column to represent not the date the earnings were released, but the date of the trading day following this. Here we can use the **get.trade.dates** function which takes in a vector of dates, and returns an equal-sized vector of the next valid trade dates for these. We will first flatten the matrix, then add a trade.date column.

```
earnings.surprise <- as.data.frame(as.table(earnings.surprise))
colnames(earnings.surprise) <- c("date","seccode","surprise")
earnings.surprise$trade.date <- get.trade.dates(earnings.surprise$date)
```

Now, we can calculate the returns on our close prices, skipping weekends and holidays, and flatten this too.

```
close.prices <- close.prices[rownames(close.prices) %in%
as.character(unique(get.trade.dates(rownames(close.prices)))),]
returns <- close.prices[2:nrow(close.prices),] / close.prices[1:(nrow(close.prices)-1),]
returns <- as.data.frame(as.table(returns))
colnames(returns) <- c("trade.date","seccode","return")
```

And finally, merge the two.

```
combined <- merge(earnings.surprise, returns, by=c("trade.date","seccode"))
```

With the data in place from QAD, analysis using other R libraries could now begin.