Utilities to Compare Time Series on Different Databases (for equality)

March 7, 2012

1 Introduction

The code from the vignette that generates this guide can be loaded into an editor with *edit(vignette("TScompare"))*. This uses the default editor, which can be changed using *options()*. It should be possible to view the pdf version of the guide for this package with *print(vignette("TScompare"))*.

Once R is started, the functions in this package are made available with

```
> library("TScompare")
```

This will also load required packages *TSdbi*, *DBI*, *methods*, *tframePlus*, *zoo*, and *tseries*.

The main purpose of this package is to compare pairs of series on two database. These sereis might have the same name, but for generality the main function, *TScompare*, is set up to use name pairs. The pairs to compare are indicated by a matrix of strings with two columns. Alternately, it would be possible to compare pairs on the same database but, other than for testing or demonstration purposes, this would not make sense unless the names are different.

The connections are established using other TSdbi packages such as TSMySQL, TSpadi, etc. It will be necessary to establish two database connections, so it will also be necessary to load the database specific packages. In this vignette, examples will use *TShistQuote*, *TSMySQL* and *TSSQLite*.

```
> library("TShistQuote")
> library("TSMySQL")
> library("TSSQLite")
```

1.1 Examples using constructed database

To provide simple examples, *TShistQuote* is used to extract some series from the web and save them on local databases. First local MySQL andSQLite test databases are created. The next small section of code determines the username

and password, or sets them to an empty string if they are to be taken from a configuration file, and sets up the databases. See the vignettes for *TSMySQL* and *TSSQLite* for more details and other options to set the username and password.

```
>   user    <- Sys.getenv("MYSQL_USER")
>   if ("" != user) {
        host    <- Sys.getenv("MYSQL_HOST")
        if ("" == host)          host <- Sys.info()["nodename"]
        passwd  <- Sys.getenv("MYSQL_PASSWD")
        if ("" == passwd)   passwd <- NULL
       }
>   con <-  if ("" == user) dbConnect("MySQL", dbname="test") else
      dbConnect("MySQL", dbname="test", username=user, password=passwd, host=host)
>   source(system.file("TSsql/CreateTables.TSsql", package = "TSdbi"))
>   dbDisconnect(con)

> con <- dbConnect("SQLite", dbname="test")
> source(system.file("TSsql/CreateTables.TSsql", package = "TSdbi"))
> dbDisconnect(con)
```

Now a TS connection to the database is established.

```
>   con1 <-  if ("" == user) TSconnect("MySQL", dbname="test") else
      TSconnect("MySQL", dbname="test", username=user, password=passwd, host=host)
>   con2 <- TSconnect("SQLite", dbname="test")
```

Next a connection to yahoo is used to get some series and write them to the local test database. (See the vignette for *TShistQuote* for more examples of reading series from the web.) *TSreplace* is used because *TSput* will fail if the series already exisits.

```
>   yahoo <- TSconnect("histQuote", dbname="yahoo")
>   x <- TSget("^ftse", yahoo)
>   TSreplace(x, serIDs="ftse", Table="B", con=con1)

[1] TRUE

>   TSreplace(x, serIDs="ftse", Table="B", con=con2)

[1] TRUE

>   x <- TSget("^gspc", yahoo)
>   TSreplace(x,  serIDs="gspc", Table="B", con=con1)

[1] TRUE

>   TSreplace(x,  serIDs="gspc", Table="B", con=con2)

[1] TRUE
```

```
>   x <- TSget("ibm", con=yahoo, quote = c("Close", "Vol"))
>   TSreplace(x, serIDs=c("ibmClose", "ibmVol"), Table="B", con=con1)

[1] TRUE

>   TSreplace(x, serIDs=c("ibmC",     "ibmV"),   Table="B", con=con2)

[1] TRUE
```

Now to do a comparison

```
>   ids <- AllIds(con1)
>   print(ids)

[1] "ftse"     "gspc"     "ibmClose" "ibmVol"
```

If the second database has the same names then ids can be made into a matrix with identical columns.

```
>   ids <- cbind(ids, ids)
>   eq   <- TScompare(ids, con1, con2, na.rm=FALSE)
>   print(summary(eq))

4  of  4 are available on con1.
2  of  4 are available on con2.
2  of  2 remaining have the same window.
2  of  2 remaining have the same window and values.

>   eqrm <- TScompare(ids, con1, con2, na.rm=TRUE)
>   print(summary(eqrm))

4  of  4 are available on con1.
2  of  4 are available on con2.
2  of  2 remaining have the same window.
2  of  2 remaining have the same window and values.
```

Since names are not identical the above indicates discrepancies, which are resolves by indicating the corresponding name pairs.

```
>   ids <- matrix(c("ftse","gspc","ibmClose", "ibmVol",
                    "ftse","gspc","ibmC", "ibmV"),4,2)
>   ids

     [,1]       [,2]
[1,] "ftse"     "ftse"
[2,] "gspc"     "gspc"
[3,] "ibmClose" "ibmC"
[4,] "ibmVol"   "ibmV"
```

```
>   eq   <- TScompare(ids, con1, con2, na.rm=FALSE)
>   print(summary(eq))

4  of  4 are available on con1.
4  of  4 are available on con2.
4  of  4 remaining have the same window.
4  of  4 remaining have the same window and values.

>   eqrm <- TScompare(ids, con1, con2, na.rm=TRUE)
>   print(summary(eqrm))

4  of  4 are available on con1.
4  of  4 are available on con2.
4  of  4 remaining have the same window.
4  of  4 remaining have the same window and values.
```