Introduction to Revolution R Enterprise Module 2: Getting Data to RRE

REVOLUTION
ANALYTICS

Overview

This section picks up in establishing the Data Source to access information contained in an xdf file and relaying that information to ScaleR. In this section we will go through the process of establishing the proper physical commands for creating a Data Source, as well as using the Data Source to execute ScaleR function commands.

Converting Large Data Sets to xdf File Format

Again, data has already been converted to xdf file format for the purposes of this course.

- Two options:
  - If data is small enough, may upload the file into the environment to create a memory-based object
  - If data is too large to upload into memory, then use ScaleR to convert data from original file format to an .XDF file (for exploratory or iterative analysis)
  - If data is too large to upload into memory, then use ScaleR to create a data source (for occasional analysis)

Example: Importing Bank Data

Using the Bank data, let's convert it to an XDF file to be more efficient and so that we can conduct analyses using Hadoop.

- First, specify the file path on the local directory & an output XDF file in the proper local directory

```
infile <- file.path("data", "bank-full.csv")
outfile <- file.path("data", "BankXDF.xdf")
```

- Import the data using rxImport, We use colClasses to specify the class fo each variable upon import. Our end result is and xdf file containing the data and a RxXdfData Source pointing to the xdf file.

```
colClasses <- c("integer", rep("factor", 4), "integer", rep("factor", 3), "integer",
    "factor", rep("integer", 4), rep("factor", 2))

names(colClasses) <- c("age", "job", "marital", "education", "default", "balance",
```

“Importing” Churn Data Xdf

Moving forward, we will also want to use the Churn data set, named ChurnXDF.xdf. Since this file is already in the xdf file format we can simply create a pointer to it.

```r
ChurnXDF <- file.path("data", "ChurnData.xdf")
rxGetInfo(ChurnXDF, numRows = 6)
```

```
## File name: /AcademyR/Revolution_Course_Materials/modules/IntroToR/Big_Data_Challenges_and_Data_
## Number of observations: 10000
## Number of variables: 28
## Number of blocks: 1
## Compression type: zlib
## Data (6 rows starting with row 1):
...
```

Creating a Data Source

Instead of importing a file and creating an xdf file containing the data, we have the option of creating a data source.

A data source can be thought of as a map pointing to the data. When calling functions the user treats the data source in the same way as she would using a standard data set in ScaleR. The only difference is that the data source then tells ScaleR where to find the data. ScaleR compiles the data and executes the computation on the data set.

Creating a Data Source: The Data Source

To create the data source to a csv file use the following commands

```
infile <- file.path("data", "bank-full.csv")
BankDS <- RxTextData(file = infile)

rxGetInfo(BankDS, numRows = 6)
```

```
## File name: data/bank-full.csv
## Data Source: Text
## Data (6 rows starting with row 1):
##   age          job marital education default balance housing loan contact
## 1  58   management married  tertiary      no    2143     yes   no unknown
## 2  44   technician  single secondary      no      29     yes   no unknown
...
```

Data Source Constructors

Source Data	Data Source Constructor
Text	RxTextData
SAS	RxSasData
SPSS	RxSpssData
Database	RxOdbcData
Teradata Database	RxTeradata

Exercise: Create a Data Source

In this exercise, create a data source for the BankData xdf data (BankXDF.xdf)

Exercise: Solution

This is a relatively easy solution. Simply execute the RxXdfData function command specifying the proper location of the file:

```
infile <- file.path("data", "BankXDF.xdf")
BankDS <- RxXdfData(file = infile)

rxGetInfo(BankDS, getVarInfo = TRUE, numRows = 6)
```

```
## File name: /AcademyR/Revolution_Course_Materials/modules/IntroToR/Big_Data_Challenges_and_Data_
## Number of observations: 45211
## Number of variables: 17
## Number of blocks: 5
## Compression type: zlib
## Variable information:
...
```

```
rxSummary(~., BankDS)
```

```
## Rows Read: 10000, Total Rows Processed: 10000, Total Chunk Time: 0.004 seconds
## Rows Read: 10000, Total Rows Processed: 20000, Total Chunk Time: 0.009 seconds
## Rows Read: 10000, Total Rows Processed: 30000, Total Chunk Time: 0.006 seconds
## Rows Read: 10000, Total Rows Processed: 40000, Total Chunk Time: 0.007 seconds
## Rows Read: 5211, Total Rows Processed: 45211, Total Chunk Time: 0.005 seconds
```

Recap

Let's review some of the concepts:

- What options do you have for using ScaleR on non-xdf files?
- What ScaleR function(s) can you use to implement each option?

Thank you

Revolution Analytics is the leading commercial provider of software and support for the popular open source R statistics language.

www.revolutionanalytics.com, 1.855.GET.REVO, Twitter: @RevolutionR