Pivotal™

PivotalR 0.1 Quick Start

Rev: A01

Date: 6/4/13

Introduction

PivotalR enables users of R, the most popular open source statistical programming language and environment to interact with the Greenplum Database (GPDB), PivotalHD, and HAWQ for Big Data analytics.

Using PivotalR, you can develop, refine, and deploy R scripts that leverage the parallelism and scalability of the database. You can also leverage in-database analytics libraries to operate on big data sets that do not fit in R memory. You can do this without knowledge of SQL.

PivotalR 1.0 provides core R infrastructure and data frame functions and over fifty R analytical functions that leverage in-database execution. Some of the key analytical functions are as follows:

- Data Connectivity – db.connect, db.disconnect, db.Rquery
- Data Exploration – db.data.frame, subsets
- R language features – dim, names, min, max, nrow, ncol, summary etc
- Reorganization Functions – merge, by (group-by), samples
- Transformations – as.factor, null replacement
- Algorithms – linear regression and logistic regression wrappers for MADlib,
- PivotalR Specific - content, preview

Setting up PivotalR

This section contains the following information:

- Prerequisites
- Installing PivotalR
- Loading PivotalR
- Getting Help

Prerequisites

Check that you have met the following prerequisites before you download the package:

- R 2.11 and later (download R from http://www.r-project.org)
- GPDB 4.2.0 and higher; HAWQ 1.0 or higher; Postgres 9.0 or higher
- MADlib 0.6 or higher

Installing PivotalR

You can install the package using one of the following methods:
From the PivotalR repository:

1. Select https://github.com/madlib-internal/PivotalR/archive/master.zip
2. Open the PivotalR Installer.
3. Select Packages -> Install the packages from local zip file.
4. Browse to the folder containing PivotalR-master.zip.

From the CRAN (Comprehensive R Archive Network)

1. Open R GUI
2. Type the following at the command line
   install.packages(“PivotalR”)
3. If prompted, select a CRAN mirror.

Loading PivotalR

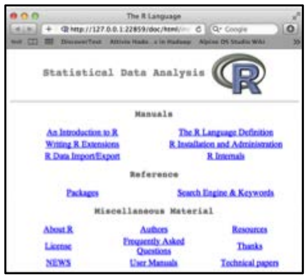
You must load the PivotalR package each time:

1. Open the PivotalR GUI.
2. Type the following at the command line:
   library(PivotalR)

Getting Help

You can access help from the PivotalR GUI:

1. Type the following at the command line:
   help.start()
   This launches the R documentation page as follows:



Statistical Data Analysis

Manuals

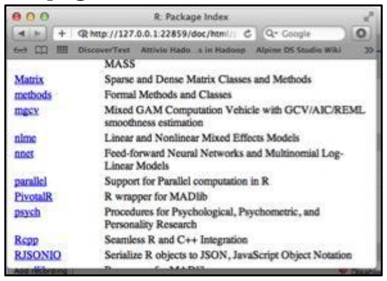
An Introduction to R
The R Language Definition
Writing R Extensions
R Installation and Administration
R Data Import/Export
R Internals

Reference

Packages
Search Engine & Keywords

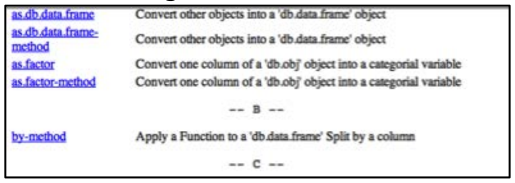
Miscellaneous Material

About R
Authors
Resources
License
Frequently Asked Questions
Thanks
NEWS
User Manuals
Technical papers



	MASS
Matrix	Sparse and Dense Matrix Classes and Methods
methods	Formal Methods and Classes
mgcv	Mixed GAM Computation Vehicle with GCV/AIC/REML smoothness estimation
nlme	Linear and Nonlinear Mixed Effects Models
nnet	Feed-forward Neural Networks and Multinomial Log-Linear Models
parallel	Support for Parallel computation in R
PivotalR	R wrapper for MADlib
psych	Procedures for Psychological, Psychometric, and Personality Research
Rcpp	Seamless R and C++ Integration
RJSONIO	Serialize R objects to JSON, JavaScript Object Notation

2. Click on Packages and link to PivotalR:



as.db.data.frame	Convert other objects into a 'db.data.frame' object
as.db.data.frame-method	Convert other objects into a 'db.data.frame' object
as.factor	Convert one column of a 'db.obj' object into a categorial variable
as.factor-method	Convert one column of a 'db.obj' object into a categorial variable
	-- B --
by-method	Apply a Function to a 'db.data.frame' Split by a column
	-- C --

   For more information, see the resources section.

Using PivotalR

You can use PivotalR to connect to GPDB or HAWQ, establish database data frames (virtual R data frames) to tables, and use the standard R syntax to view and manipulate data.

Connecting to GPDB or HAWQ

Using PivotalR you can create and use multiple database connections. You can also connect to different databases within the same session:

Example:

db.connect(host = "hostname", user = “username”, dbname = “database_name”, password = "your password", port = 5432, madlib = "madlib")
Created a connection to database with ID 2
[1] 2

This command creates a connection id for each database connection. Use this connection id in the other functions when specifying which database to use. When disconnecting, you must use the connection id.
***Note**:* All database connections are automatically closed when the R session is closed.

Example:

db.disconnect(conn.id = 2)
Connection 2 is disconnected!
[1] TRUE

Creating a Database Data Frame

A database data frame is an R object that represents a GPDB, HAWQ, or Postgres table or view. The database data frame stores the link between the database table and the R environment. The object is initialized during creation and contains information including the total number of rows and the table column names. Only the initial values of rows and columns persist and are stored in the database data frame object. The table data does not reside in the R system, To create a database data frame, use the following command:

x <- db.data.frame("table_name", key = "table primary key")

In this command, "table_name" is a table in the current database for the connection.

Add the conn.id parameter if the table resides in another database:

y <- db.data.frame("table_name", key = "table primary key", conn.id = connection id)

Note: It is optional to specify a key. If you specify a key, you can use an expression such as x[1:2,]. Otherwise, you need to explicitly specify the column name such as x[x$id == 1 | x$id == 2,].

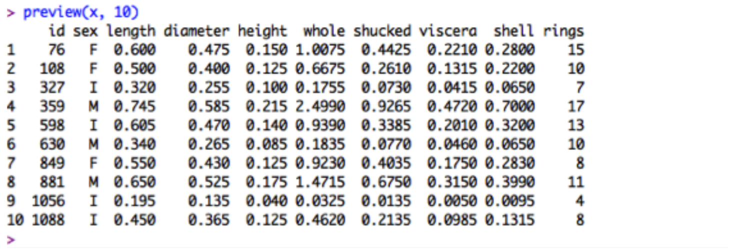
The column names and dimension of the table can be obtained using the standard R commands:

- *names(x)*
- *dim(x)*

Previewing Data in a Database Data Frame

The database data frame object is a pointer with basic metadata information. PivotalR provides the 'preview' function to preview data in the table directly in the R console.

Example:

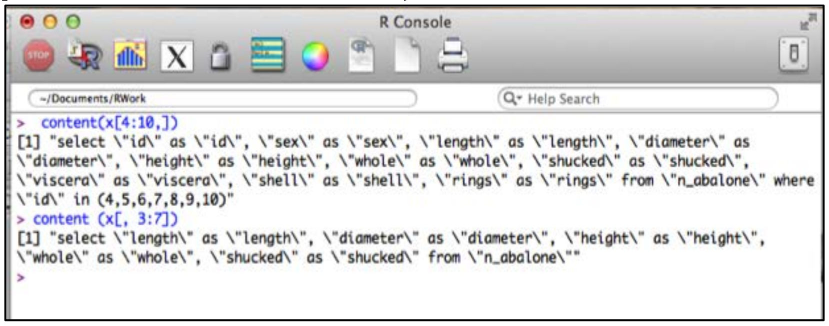


```
> preview(x, 10)
     id sex length diameter height  whole shucked viscera  shell rings
1    76   F  0.600    0.475  0.150 1.0075  0.4425  0.2210 0.2800    15
2   108   F  0.500    0.400  0.125 0.6675  0.2610  0.1315 0.2200    10
3   327   I  0.320    0.255  0.100 0.1755  0.0730  0.0415 0.0650     7
4   359   M  0.745    0.585  0.215 2.4990  0.9265  0.4720 0.7000    17
5   598   I  0.605    0.470  0.140 0.9390  0.3385  0.2010 0.3200    13
6   630   M  0.340    0.265  0.085 0.1835  0.0770  0.0460 0.0650    10
7   849   F  0.550    0.430  0.125 0.9230  0.4035  0.1750 0.2830     8
8   881   M  0.650    0.525  0.175 1.4715  0.6750  0.3150 0.3990    11
9  1056   I  0.195    0.135  0.040 0.0325  0.0135  0.0050 0.0095     4
10 1088   I  0.450    0.365  0.125 0.4620  0.2135  0.0985 0.1315     8
>
```

The example shows a preview of 10 rows from the table mapped to database data frame object X.

Viewing SQL pushed down to the Database

PivotalR provides a specialized 'content' function to view the SQL string generated by operations performed on the DB data frame object.



```
>  content(x[4:10,])
[1] "select \"id\" as \"id\", \"sex\" as \"sex\", \"length\" as \"length\", \"diameter\" as
\"diameter\", \"height\" as \"height\", \"whole\" as \"whole\", \"shucked\" as \"shucked\",
\"viscera\" as \"viscera\", \"shell\" as \"shell\", \"rings\" as \"rings\" from \"n_abalone\" where
\"id\" in (4,5,6,7,8,9,10)"
> content (x[, 3:7])
[1] "select \"length\" as \"length\", \"diameter\" as \"diameter\", \"height\" as \"height\",
\"whole\" as \"whole\", \"shucked\" as \"shucked\" from \"n_abalone\""
>
```

Executing Operations on Database Data Frames

PivotalR supports the common subset operations on data frames including:

- Selecting a subset of rows and columns on specific logical conditions.
  - *content(y[,1:2])*
  - *content(y[y$nation < 3, 1:2])*
- Pivoting categorical variables via as.factor:
  t$nation <- as.factor(t$nation)
- Replacing null values
  - *y[is.na(y$nation),"nation"] <- 3*

- Merging data frames
    - *m <- merge(y, z, by = c("id", "sex"))*
    - *content(m)*
    - *preview(sort(m, by = "id"))*

Running In-database Analytics

You can run in-database analytics or MADlib functions:

- summary function
- *madlib.summary(s)*
- *madlib.summary(s$nation)*

- For linear regression
    - *f <- madlib.lm(interlocks ~ assets + nation, data = t)* - ***basic linear regression syntax***
    - *f <- madlib.lm(interlocks ~ assets + nation * sector | nation, data = t)* - ***with grouping by nation***
    - *f <- madlib.lm(interlocks ~ assets + as.factor(nation), data = t)* – ***directly specify which column is categorical***
- For logistic regression
    - *f <- madlib.glm(interlocks < 20 ~ assets + nation, data = t, family = "binomial")* ***– logistic regression syntax in R***

Resources

- PivotalR ZIP file download. Click here
- PivotalR Tutorial Video. Click here
- GPDB + R resources e.g. PL/R. Click here
- Questions/Feedback: Email user@madlib.net