RHadoop (Installation and Run MR example)

1. R is already available by default in the cloudera image that we installed (version 3.0.1); We just need to issue the command R in a terminal and the familiar R prompt appears.
2. For RHadoop to work, the environment variables HADOOP\_CMD and HADOOP\_STREAMING have to be defined. Insert the following at the end of /etc/profile (via e.g. sudo gedit /etc/profile; also, see the link: <https://github.com/RevolutionAnalytics/RHadoop/wiki/rmr2-settings> on the value of the two variables):
3. **export** HADOOP\_CMD=/usr/bin/hadoop
4. **export** HADOOP\_STREAMING=/usr/lib/hadoop-0.20-mapreduce/contrib/streaming/hadoop-streaming-2.0.0-mr1-cdh4.4.0.jar
5. Restart the VM for the exports to take effect using the command - sudo reboot).
6. Then, download rmr2\_3.0.0.tar.gz (rmr-3.0.0) from <https://github.com/RevolutionAnalytics/RHadoop/wiki/Downloads>
7. Open an R session with elevated rights (sudo R) and issue the below commands:
8. **install**.packages(c('Rcpp', 'RJSONIO', 'bitops', 'digest', 'functional', 'stringr', 'plyr', 'reshape2','caTools'))
9. **install**.packages("/home/cloudera/Downloads/rmr2\_3.0.0.tar.gz")
10. quit()

Additional steps to check the projects compatibility with Hadoop substrate via a battery of tests:

1. First, install libcurl-devel for the RCurl package:

sudo yum install libcurl-devel

1. Then, after sudo R:
2. install.packages("devtools")
3. library("devtools")
4. install\_github("quickcheck", "RevolutionAnalytics", subdir="pkg")
5. quit()
6. Lastly, you can start the built-in checks themselves with:

R CMD check /home/cloudera/Downloads/rmr2\_3.0.0.tar.gz

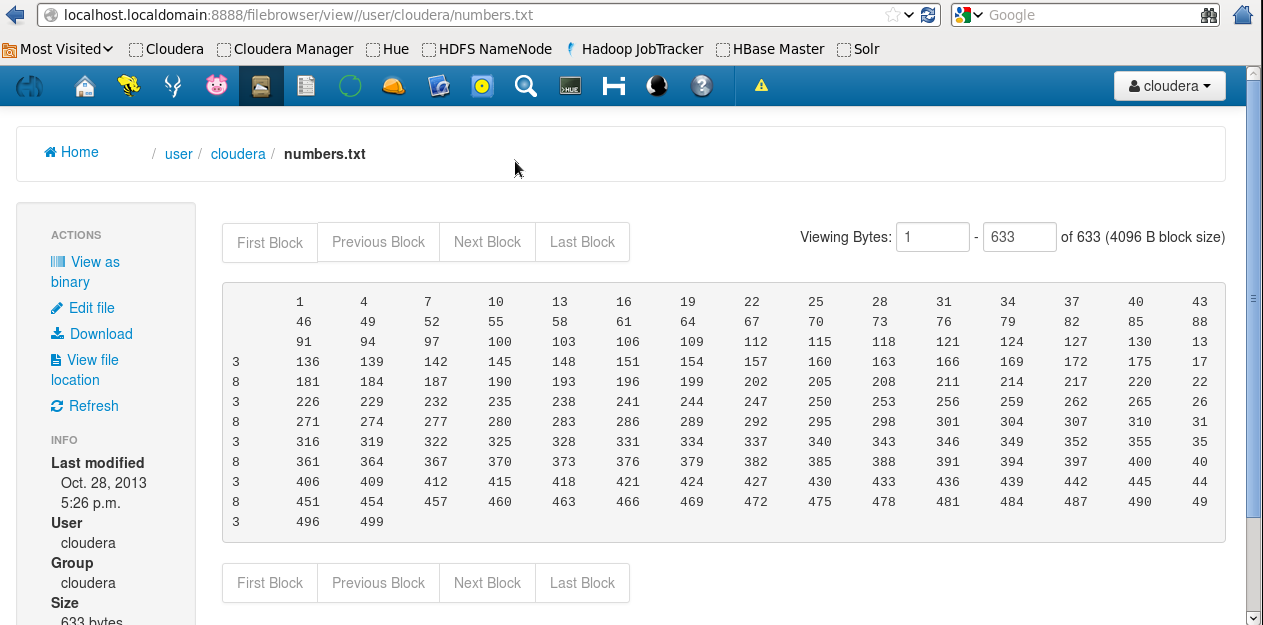
Note, however, that the test set contains 50+ Hadoop jobs, so it is quite lengthy. The results are not exactly spotless, **but all Hadoop jobs should run successfully**.

Testing RHadoop

Let's write a few numbers in human-readable form to HDFS (either normal or RStudio R session):

1. library('rmr2')
2. to.dfs(seq(from=1, to=500, by=3), format="text", output="/user/cloudera/numbers.txt")

Using Hue, we can immediately check the results:

[](http://home.mit.bme.hu/~ikocsis/notes/pics/huenums.PNG)

Finally, we can begin to write MapReduce jobs:

1. a <- to.dfs(seq(from=1, to=500, by=3), output="/user/cloudera/numbers")
2. b <- mapreduce(input=a, map=function(k,v){keyval(v,v\*v)})
3. c <- from.dfs(b)
4. d <- data.frame(key=c[["key"]], val=c[["val"]])
5. head(d[order(d$key),])

This gives the following output:

key val

1 1 1

112 4 16

157 7 49

2 10 100

13 13 169

24 16 256

Installing RStudio server:

It is quite easy to set up RStudio on the VM if you prefer to use an integrated environment over running R from the terminal. We could install the desktop version into the VM; however, you may find it more convenient to access an RStudio server from a browser on your host operating system.

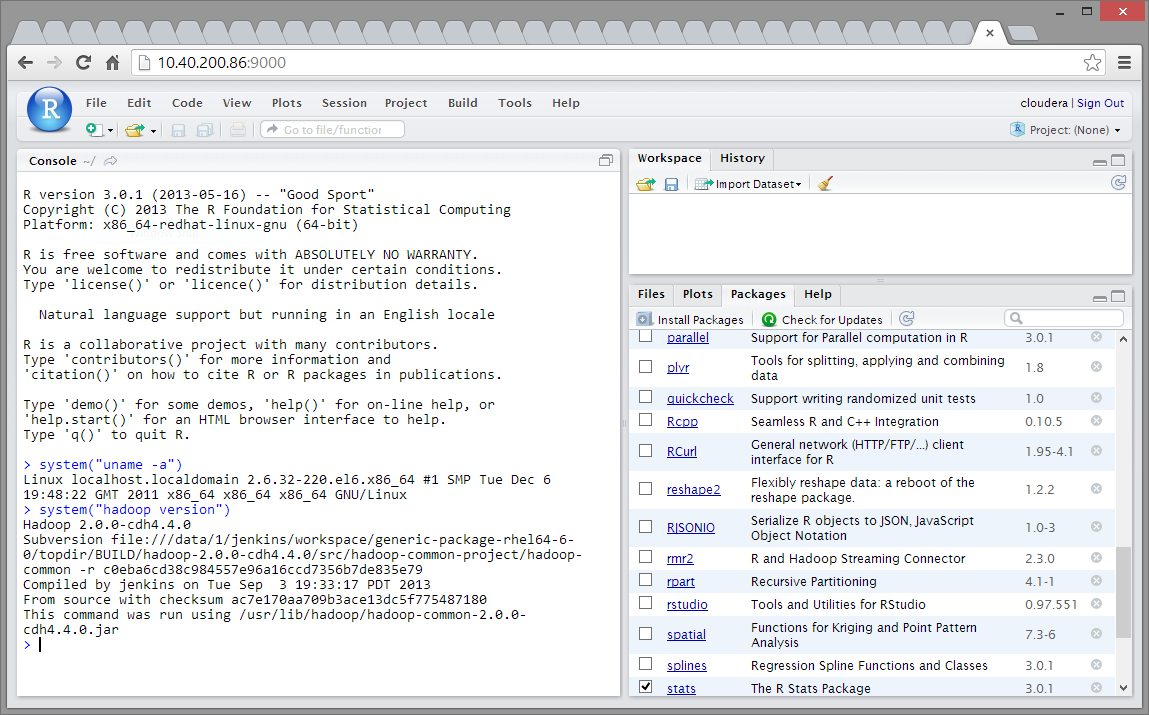
Download the rpm and install it (see the CentOS section of the [RStudio server guide](http://www.rstudio.com/ide/download/server)):

|  |  |
| --- | --- |
| 1  2 | wget http://download2.rstudio.org/rstudio-server-0.97.551-x86\_64.rpm  sudo yum install --nogpgcheck rstudio-server-0.97.551-x86\_64.rpm |

The default port RStudio server runs on is 8787; you can determine the IP address of the VM the usual way using ifconfig. Open a browser on the host OS and navigate to <VM IP address>:8787. RStudio server uses the OS for authentication; thus, you can log in with the user name and password ‘cloudera’. (Note that on the next figure port 9000 is used instead of 8787; also, the IP address is not what you would see on a default VMware NAT network. To be honest, I actually run the VM on a quite hefty remote workstation so I had to do some additional port mapping/forwarding.)

In this case you will have to take additional steps for the R session created by the server to pick up the necessary environment variables. A way to do it is to create the file Renviron.site in /usr/lib64/R/etc/ with the following content:

|  |  |
| --- | --- |
| 1  2 | export HADOOP\_CMD=/usr/bin/Hadoop  export HADOOP\_STREAMING=/usr/lib/hadoop-0.20-mapreduce/contrib/streaming/hadoop-streaming-2.0.0-mr1-cdh4.4.0.jar |

[](http://home.mit.bme.hu/~ikocsis/notes/pics/rstud_cdh.PNG)

One more thing: if you are using a non-English keyboard (layout) and use Windows as a host OS, you may experience problems with special characters that you would normally type using the AltGr button. Without going into details, Ctrl+Alt shortcuts seem to hijack the AltGr event (what on Windows is actually Ctrl+Alt) in the web application code. I am sorry to say that as of now, we know no real solution for this; either change to an English layout (-> AltGr is not necessary to type [,{,@ et al) or cope with the minor discomfort caused by the need to copy/paste certain characters.

Reference:

**http://home.mit.bme.hu/~ikocsis/notes/2013/10/28/rhadoop-sandbox-with-the-cloudera-quickstart-vm/**