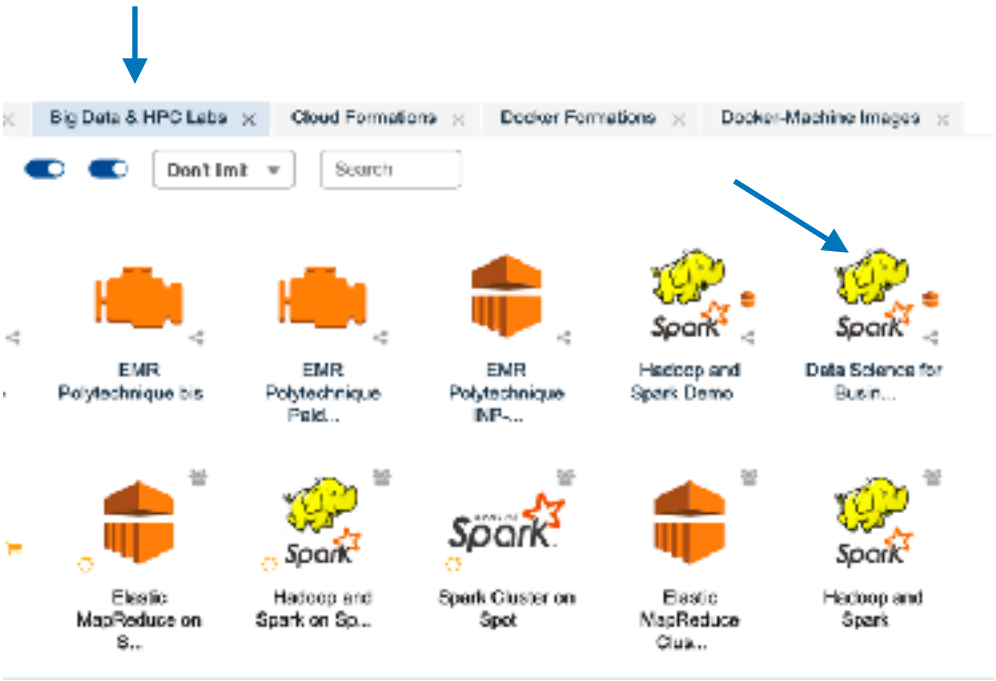
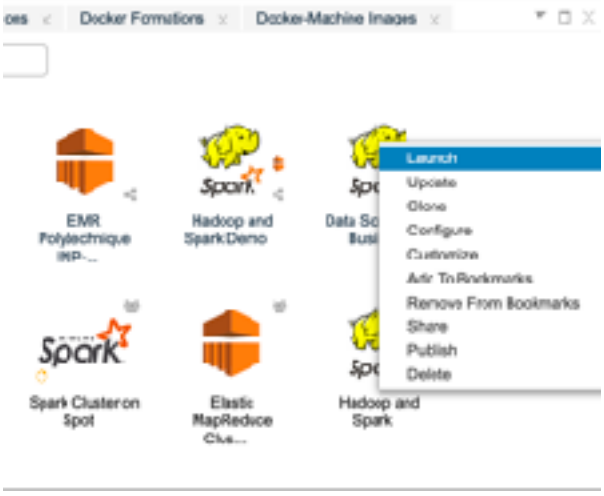
Exercises sheet n° 2: algorithms and programming in MapReduce on Hadoop

Launching your AWS cluster. Go to the RosettaHUB page (preferably by means of Chrome or Firefox)
(**https://www.rosettahub.com/welcome**) and click on Sign in to connect.

Once connected, launch 'Data Science for Business - Hadoop, Spark and HIVE ' formation. Normally this formation is automatically launched starting from 13h45. Otherwise please follow the following steps.



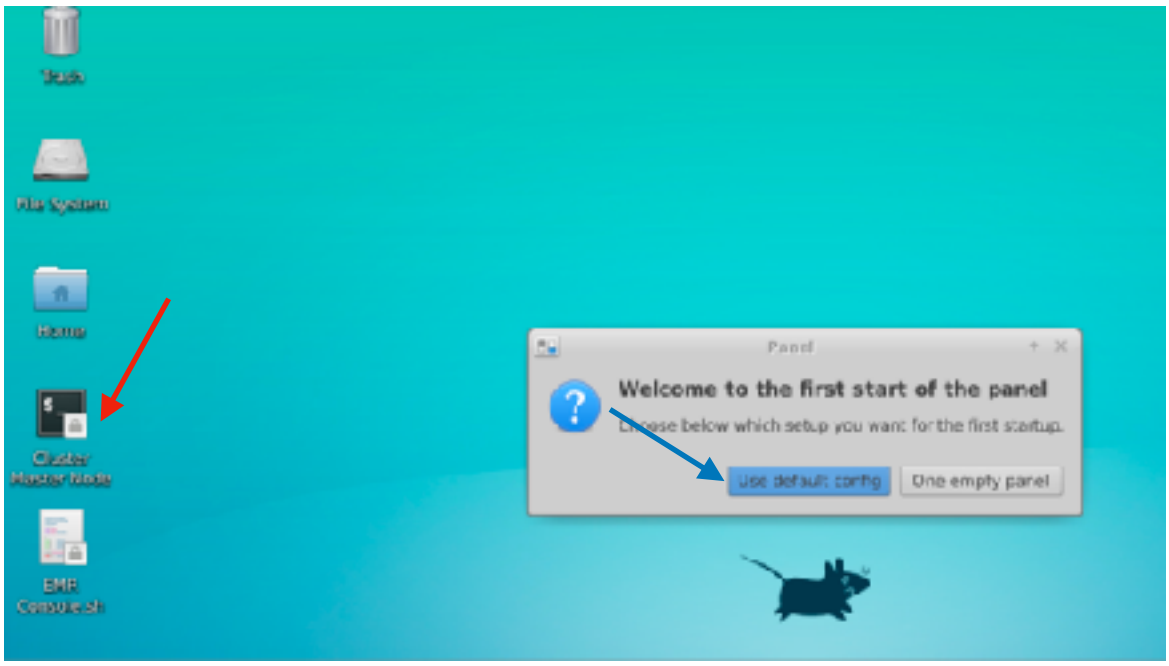
Right-click on the icon and select launch :



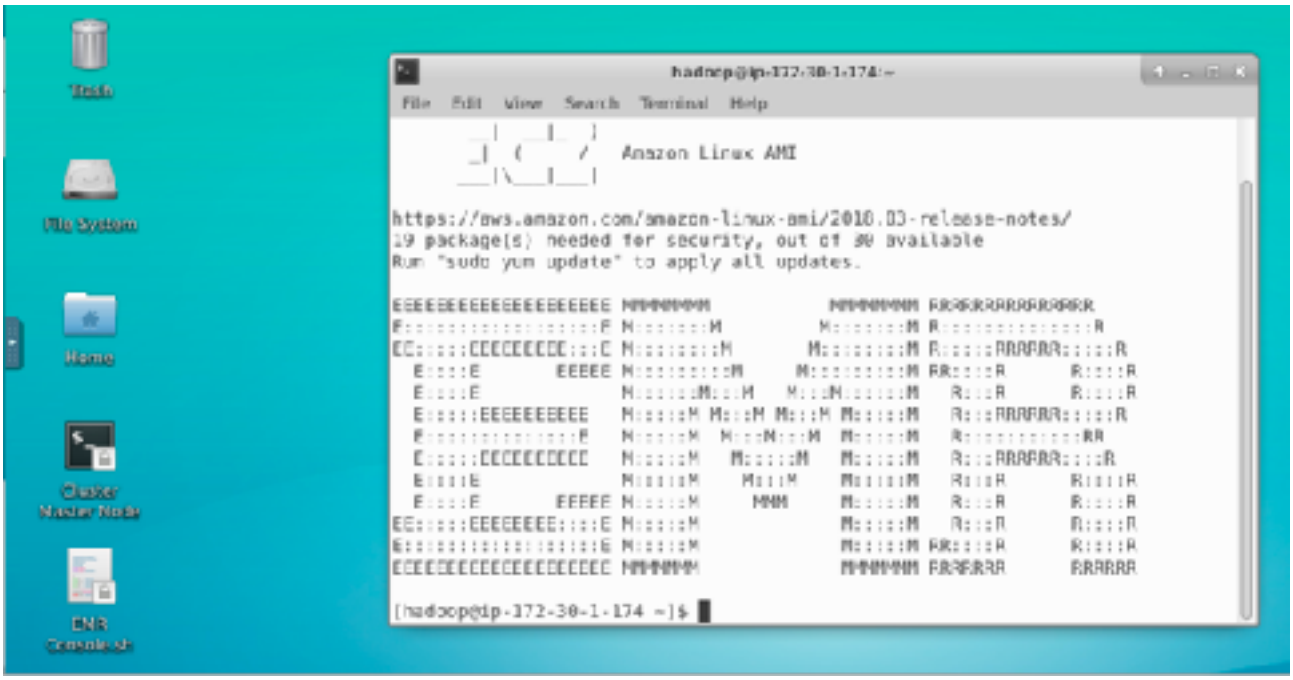
It takes a few minutes before the cluster creation is confirmed (around 5-7 minutes). Once the cluster is ready you will receive a confirmation by email, and you will see what follows.

Double-click on the new icon, you will see a new Tab as below:

Click on the Connect button, and you will see a Linux desktop, as follows

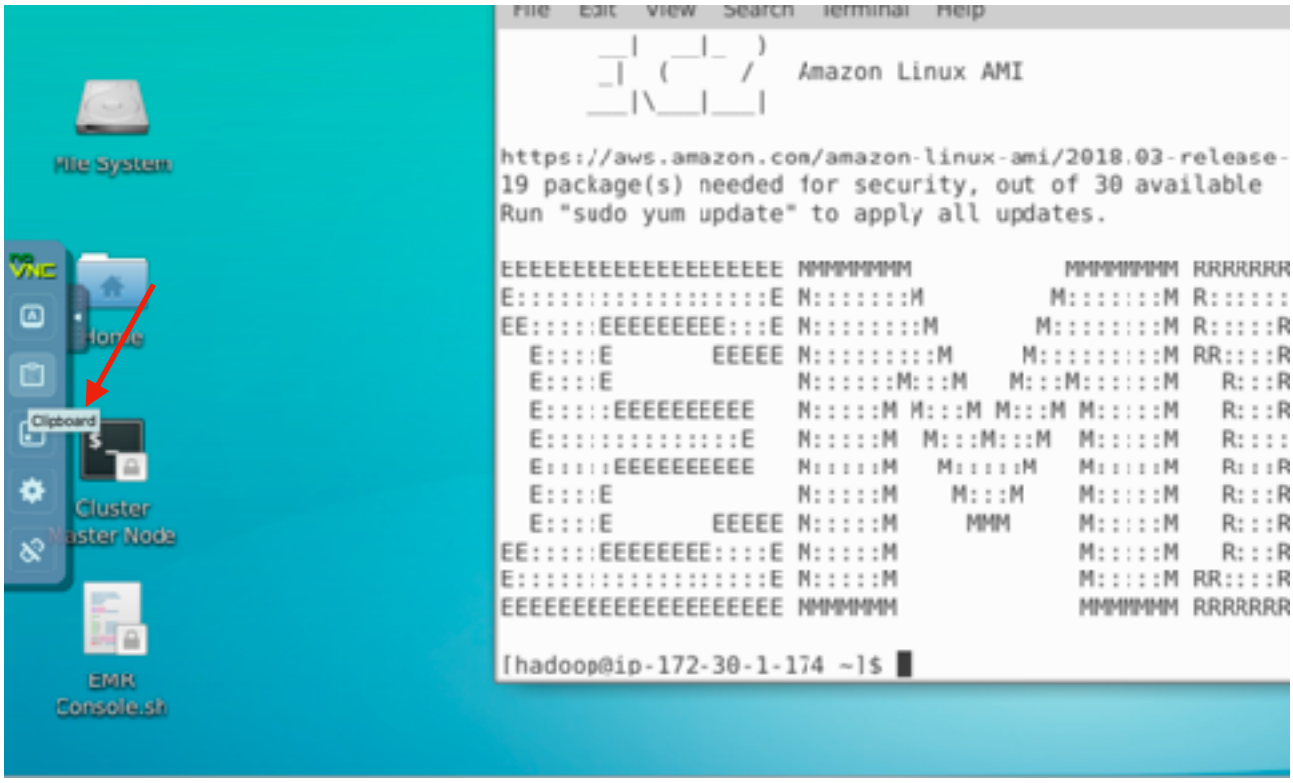


Click on 'Use default config.', then double click on the Cluster Muster Node.

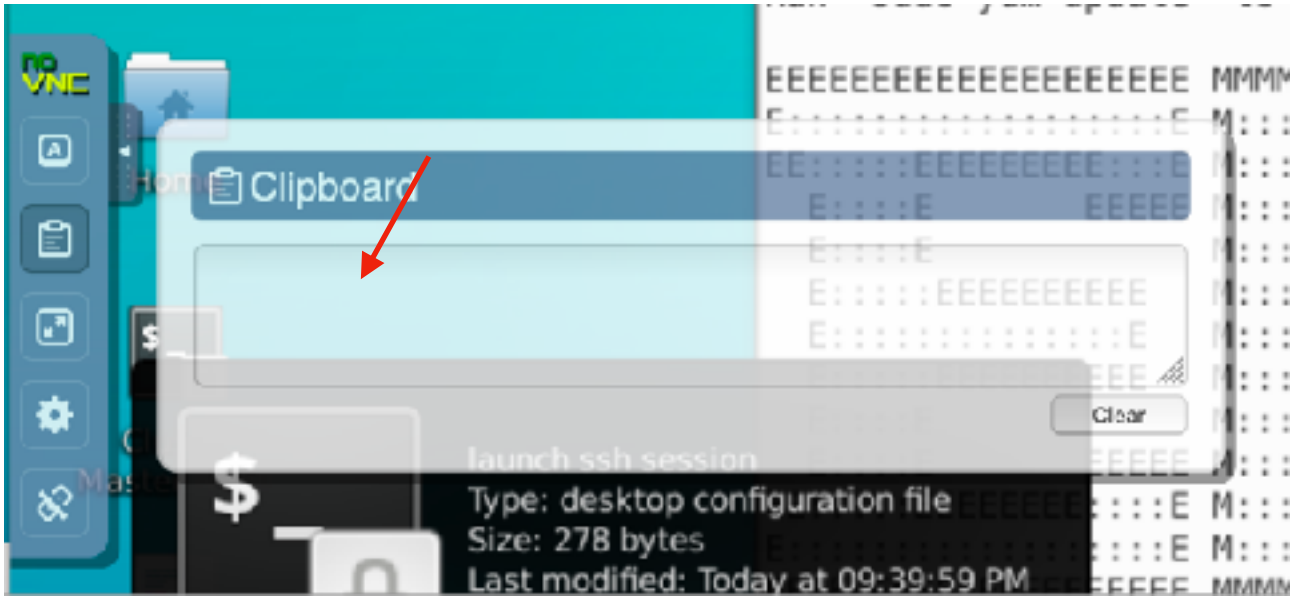


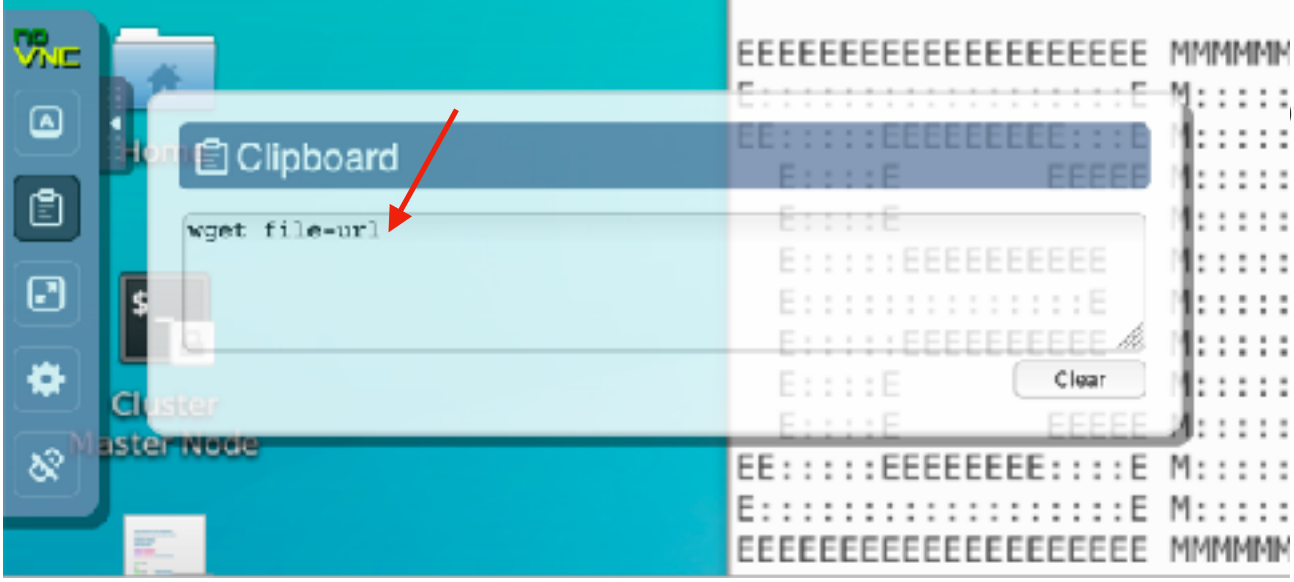
You have now access to an AWS Elastic Map Reduce cluster, and you can execute HDFS and MapReduce-related commands, as you will see.

Attention : in our lab-sessions we will need to copy-paste commands provided in the pdf sheet into the Linux command line (for instance wget command)
Since this EMR interface is not directly open to copy-paste of your operating system on you laptop, you have first to copy-paste into a clipboard space see below

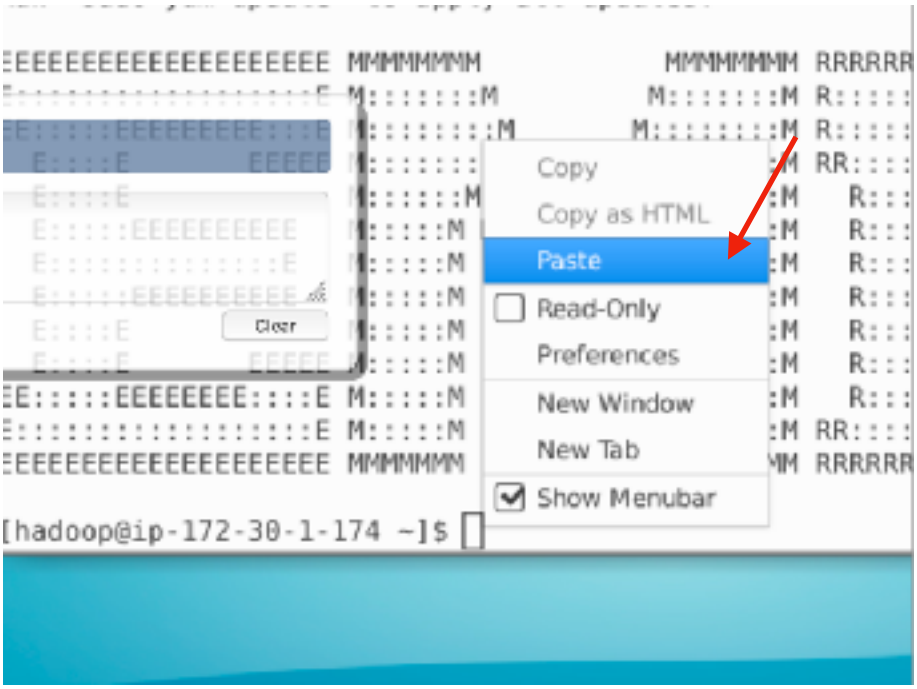


Just copy-paste first inside the clipoard opened by double-clicking the clipboard icon :

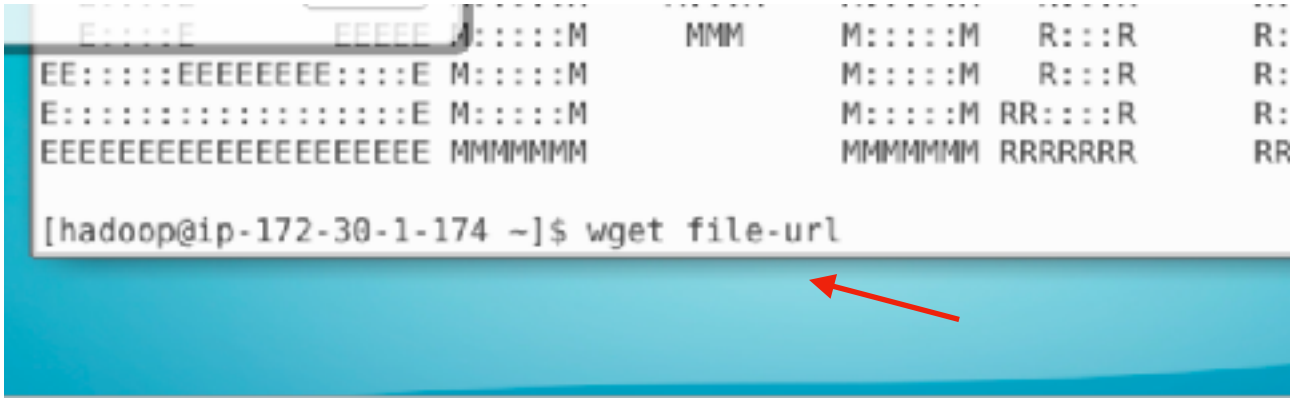




Go then on the EMR prompt in the Linux terminal, and right-click on Paste



And you will have the command in wget command in the terminal prompt:



At this point you can use HDFS (see next step) and run MapReduce jobs coded in Python, for instance the WordCount job as indicated in the following.

Your login user name is 'hadoop', and in HDFS your home is at this path: /user/hadoop

Useful commands for Linux and HDFS. See the following documents that we borrowed from the Web:

https://www.dropbox.com/s/77z7m35tfe18jyr/hadoop-hdfs-commands-cheatsheet.pdf

https://www.dropbox.com/s/06qw849h2omjke0/fwunixref.pdf

Run WordCount on the EMR Cluster.

Once connected with the AWS cluster via ssh, proceed with the following preliminary steps

1. Download the mapper, reducer and a text file in you home directory, with the following commands (copy-paste them, by following the previous procedure)

wget https://www.dropbox.com/s/5i0xbsd1hrldily/mapper.py

wget https://www.dropbox.com/s/iuu4y243466gvhl/reducer.py

wget https://www.dropbox.com/s/2f3nt4rn7t4wee1/5000-8.txt

2. Give the execution right to the .py files.

chmod +x *.py

3. You will need the absolute paths for the three files, to get the path leading your current directory use the pwd command.

4. Now you need the create a directory in you HDFS home for word count. Since your user name is 'hadoop' and your HDFS home directory is '/user/hadoop' you will use the following command to build the 'wc' directory under your HDFS home.

hdfs dfs -mkdir /user/hadoop/wc

5. Now you need to create a 'input' subdirectory containing the input fille for job

hdfs dfs -mkdir /user/hadoop/wc/input

6. Finally, you need to transfer the input file from the local file system to the HDFS file system so that MapReduce can process it.

hdfs dfs -put 5000-8.txt /user/hadoop/wc/input

7. You are ready now to launch our MapReduce job. Copy this command to a text editor, indicate the paths to needed files and then run it on the open cluster terminal. You will get several statistics. The job results are in the 'output' HDFS directory. Each reduce task has produced its own file result.

```
hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar \
-input /user/hadoop/wc/input \
-output /user/hadoop/wc/output \
-file /home/hadoop/mapper.py \
-mapper /home/hadoop/mapper.py \
-file /home/hadoop/reducer.py \
-reducer /home/hadoop/reducer.py
```

Note that we are assuming that Python programs are in your home directory /home/hadoop/ in the local file system (not on HDFS), otherwise you have to change this path.

If you want to use the combiner then use this variant (note that you re-use the reducer as a combiner)

```
hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar \
-input /user/hadoop/wc/input \
-output /user/hadoop/wc/output4 \
-file /home/hadoop/mapper.py \
-mapper /home/hadoop/mapper.py \
-file /home/hadoop/reducer.py \
-reducer /home/hadoop/reducer.py \
-combiner /home/hadoop/reducer.py
```

Of course if your combiner is not the reducer, then you have to specify both -file and -combiner parameters with the same path for the .py file including the combiner.

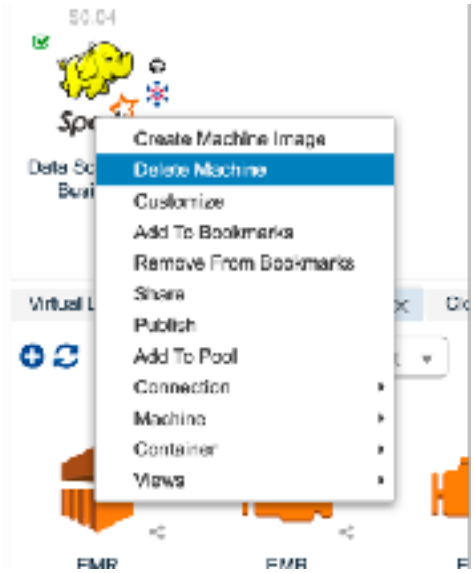
This version is to set a particular number of reduce tasks (e.g., 3 reduce tasks)

```
hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar \
-input /user/hadoop/wc/input \
```

-output /user/hadoop/wc/output \
-file /home/hadoop/mapper.py \
-mapper /home/hadoop/mapper.py \
-file /home/hadoop/reducer.py \
-reducer /home/hadoop/reducer.py \
-jobconf mapred.reduce.tasks=3

8. You can use the -ls and -cat HDFS command to, respectively, list the content of the output directory, and display one of the output file.

IMPORTANT : do not forget to eliminate the Cluster when you have finished with the lab session, by right-clicking on the Cluster icon, as follows:



Exercise 1. Encoding SQL queries in MapReduce. Consider the following simple relational schema containing informations about clients and orders they made.

Customer(cid, startDate, name)

Order(#cid, total)

Note that, for the sake of simplicity, we do not have any primary key for Order. Also assume that all the fields are mandatory.

Provide the Pyhton MapReduce encoding of the following SQL queries.

- SELECT name FROM Customer WHERE month(startDate)=7
- SELECT DISTINCT name FROM Customer WHERE month(startDate)=7
- SELECT O.cid, SUM(total), COUNT(DISTINCT total) FROM Order O GROUP BY O.cid
- SELECT C.cid, O.total FROM Customer C, Order O WHERE C.cid=O.ci

For testing the jobs use the following files (use wget *file-url* for downloading them)

https://www.dropbox.com/s/tmt6u80mkrwfjkv/Customer.txt

https://www.dropbox.com/s/8n5cbmufqhzs4r3/Order.txt

Exercise 2. Consider the above query

```
SELECT  O.cid, SUM(total), COUNT(DISTINCT total)
FROM Order O
GROUP BY O.cid
```

Is your Reduce (designed in Exercice 1 for this query) memory-bound? Put in other words, does your Reduce rely on either a list or an a hash-map (dictionary) or any other data structure whose size is not constant (and that could take an arbitrary amount of RAM to be stored at run-time)?

If so, design a Reduce that uses constant RAM memory, and that does not use any kind of data collection.