Lab 1

This introductory lab is composed of three tasks. Your final objective is to run your first Hadoop application. For this goal, you must learn how to compile the source code and produce a jar, connect to a remote machine, transfer files using FTP and submit an application on a cluster.
Please note:

- at LABINF, you have all the software you need already installed in **Linux**
- you can see **the architecture of the BigData@Polito** environment in the slides on the course web page; you need to understand the difference among
  - your local machine (LABINF PC),
  - the bigdatalab.polito.it remote gateway,
  - the Hadoop cluster with its own HDFS distributed file system

http://dbdmg.polito.it/wordpress/wp-content/uploads/2018/03/00_Cluster_BigData_2x.pdf

1. Compiling using an IDE (Eclipse)

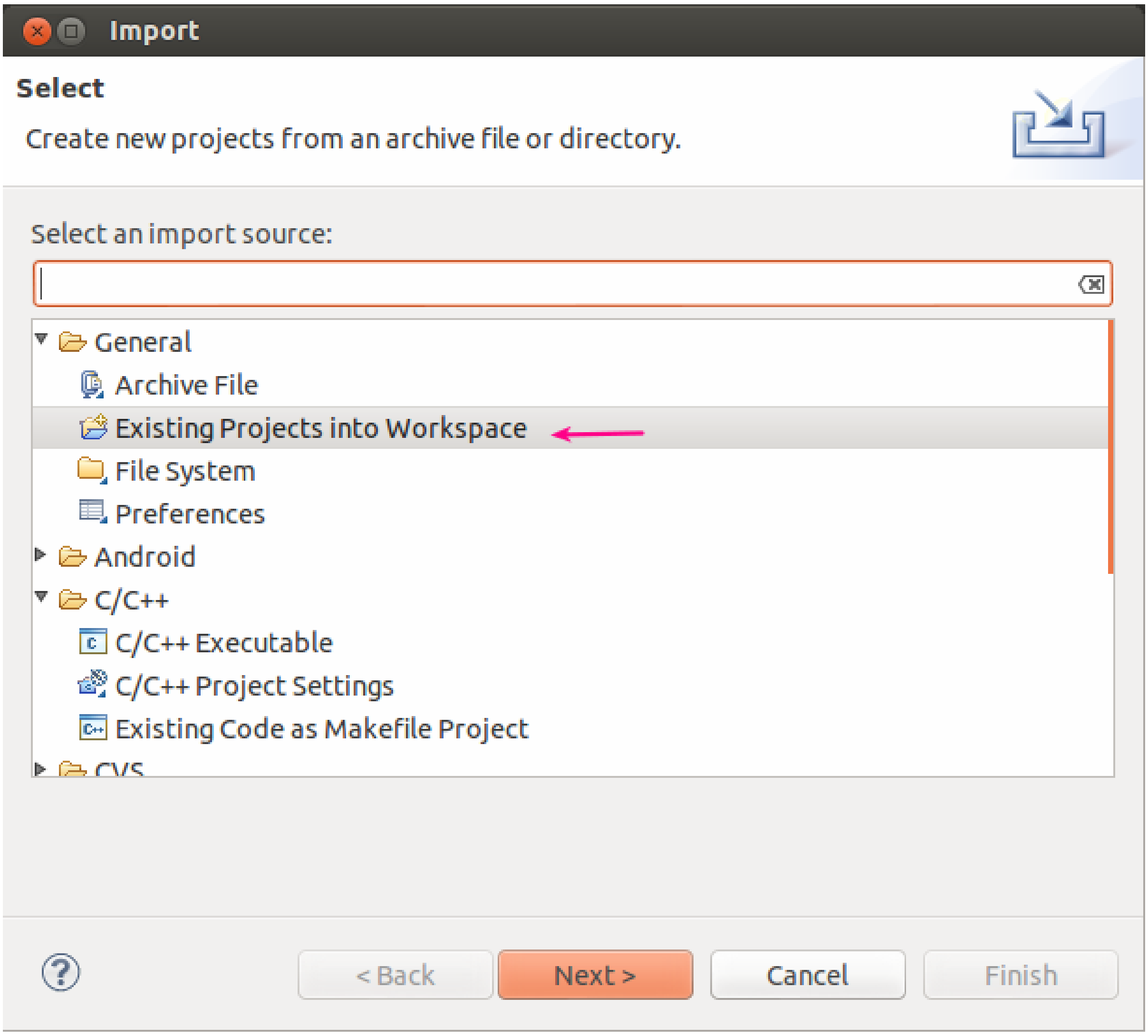
In this task, we will compile the source code of a simple Hadoop application. The easiest way would be using Maven, but here we will guide you through the manual creation of a jar instead. Remember that, if you don't use Maven, you will need to manually include the libraries every time you create a new project.
These instructions explain how to import manually the libraries (that we provide to you inside the lib/ folder of the zip). If you do not have enough free space in your home, unzip the libraries to a folder inside /tmp/, but remind that these files will be deleted at your logoff.

This first application is a simple word count.
You can use the solution available on the course web page: Lab1_BigData_with_libraries.zip

- ➔ direct link: http://dbdmg.polito.it/wordpress/wp-content/uploads/20186/03/Lab1_BigData_with_libraries.zip
- ➔ please note that Eclipse works on your local machine e generates a JAR file on your local machine (i.e., the LABINF PC)

1. Open Eclipse

2. Import the project (File -> Import… | General -> Existing project... | choose the folder where you extracted your file)



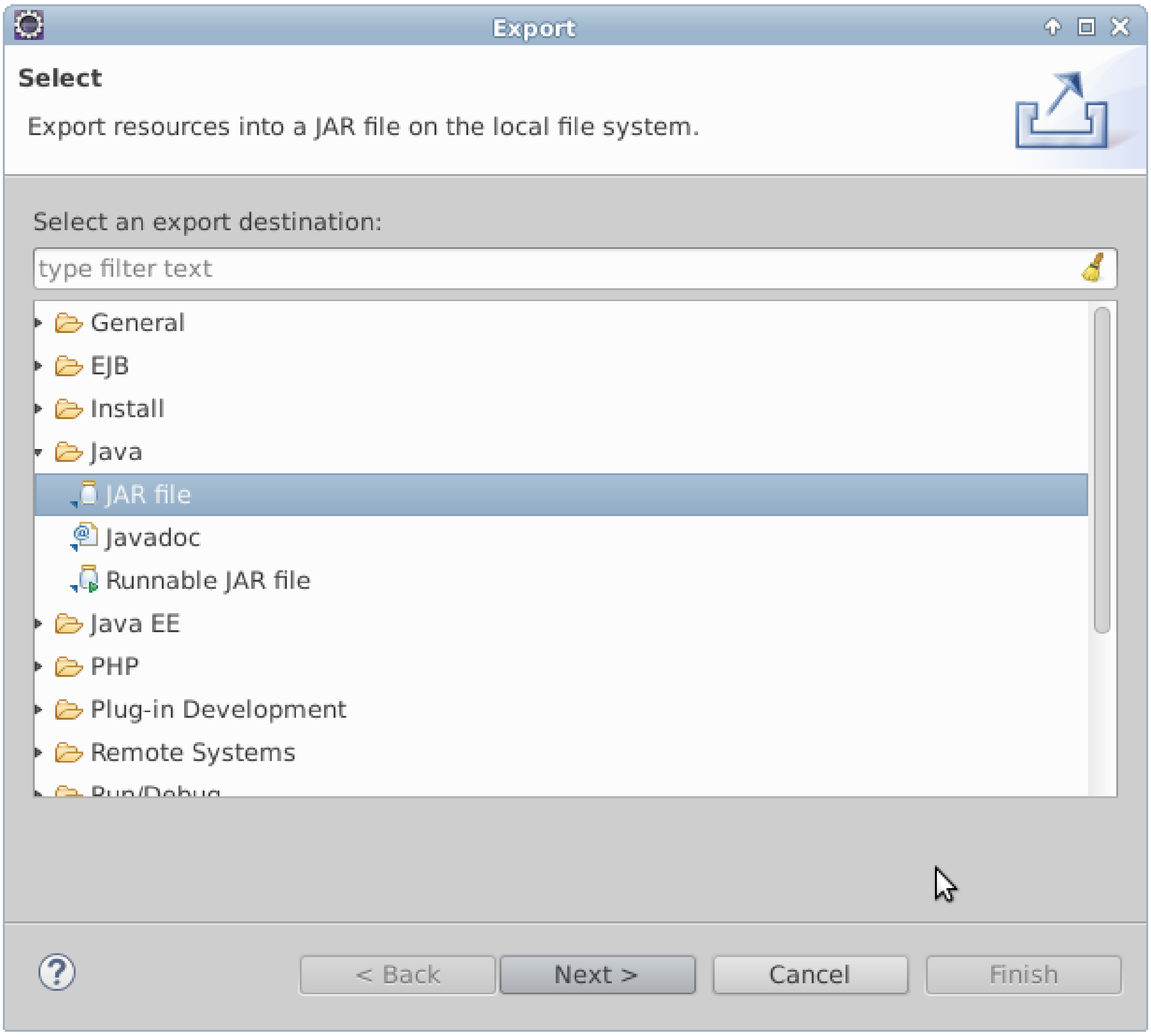
Import

Select

Create new projects from an archive file or directory.

Select an import source:

3. Have a look at the source files and the structure of the project. Where is the mapper? Where is the reducer?
4. Since we avoid using maven, we cannot count on it to generate the .jar, as we would normally do. You can instead build a jar manually using the File -> Export command, as in the following 2 figures. Remember to avoid inserting all the libraries in your jar, or you would end up producing a fat jar heavy to transfer. We need these libraries locally to compile, but they are already present in the classpath of the cluster and there is no need to include them in the jar again.



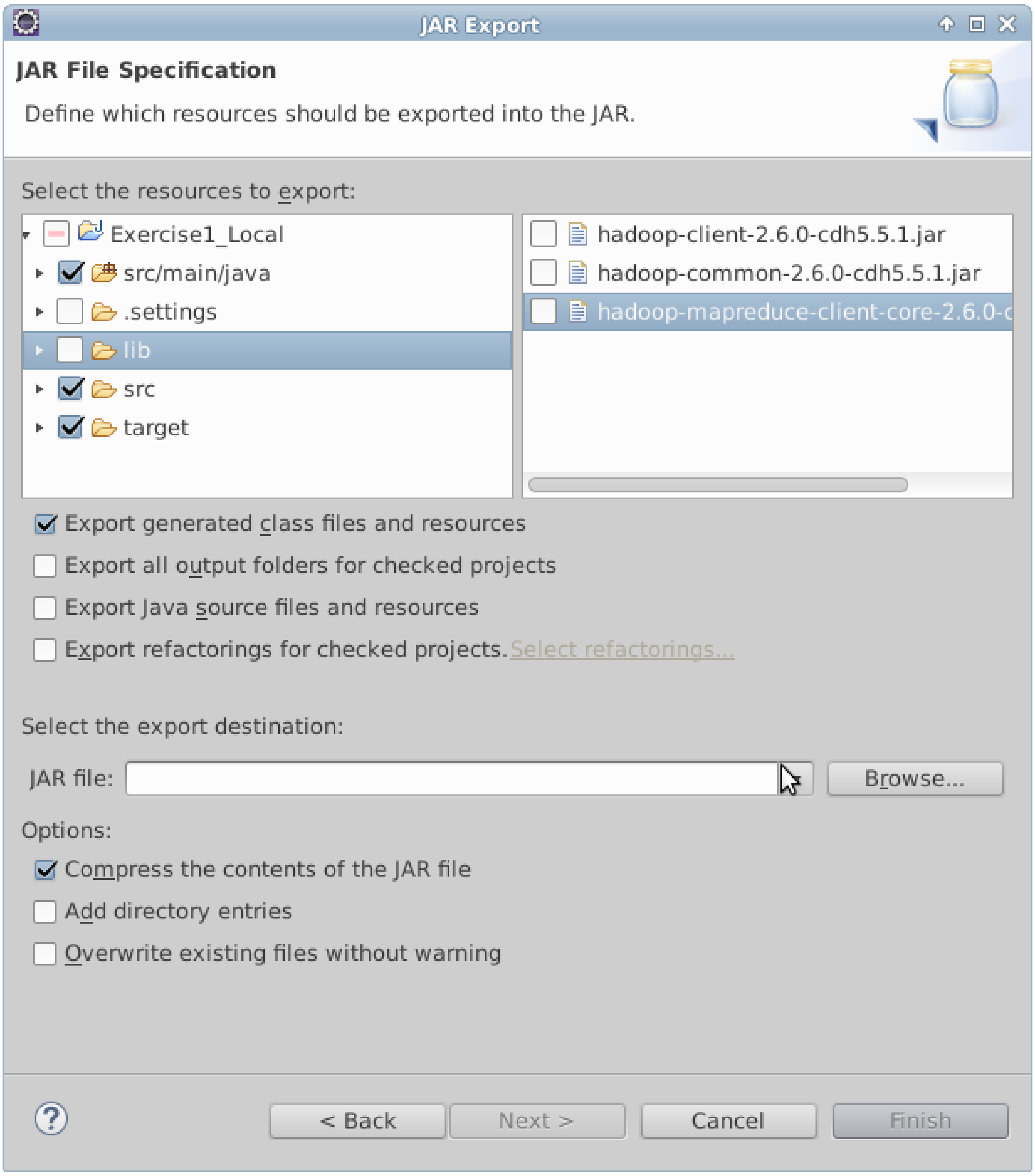
Export

Select

Export resources into a JAR file on the local file system.

Select an export destination:

type filter text



JAR Export

JAR File Specification

Define which resources should be exported into the JAR.

Select the resources to export:

- ▣ Exercise1_Local
  - ☑ src/main/java
  - □ .settings
  - □ lib
  - ☑ src
  - ☑ target

- □ hadoop-client-2.6.0-cdh5.5.1.jar
- □ hadoop-common-2.6.0-cdh5.5.1.jar
- □ hadoop-mapreduce-client-core-2.6.0-c

☑ Export generated class files and resources

□ Export all output folders for checked projects

□ Export Java source files and resources

□ Export refactorings for checked projects. Select refactorings...

Select the export destination:

JAR file:

Options:

☑ Compress the contents of the JAR file

□ Add directory entries

□ Overwrite existing files without warning

Specify the name of the output jar file (e.g., set the JAR file textbox to Exercise1-1.0.0.jar).

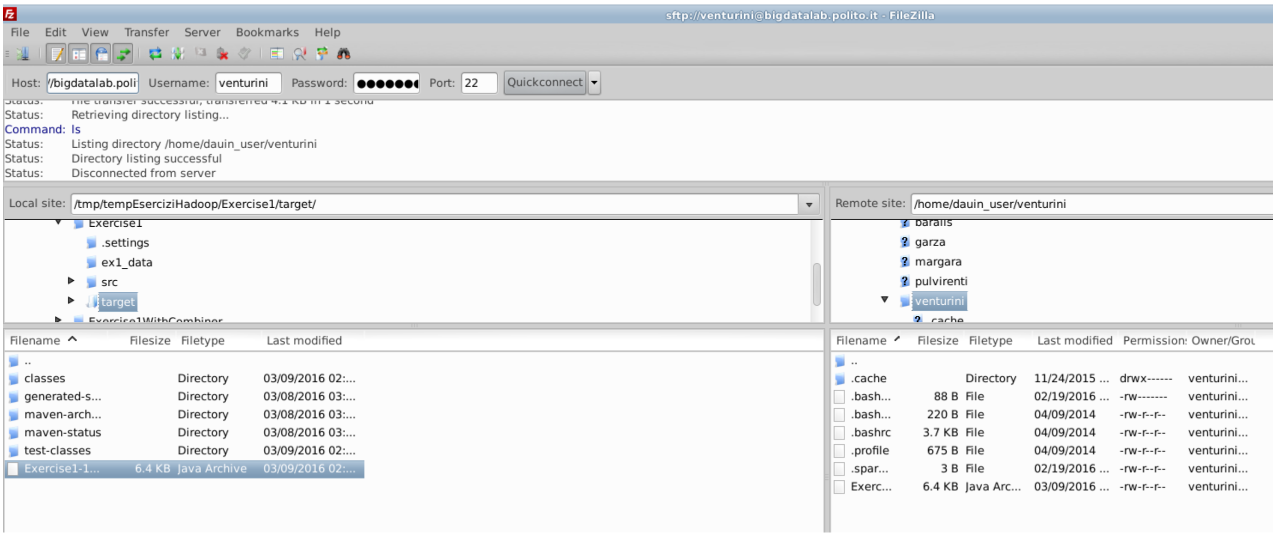
2. Transfer files with Filezilla

The objective of this task is to transfer some files from the local machine (i.e. the one you're working on right now) to a remote machine (e.g. the BigDataLab gateway). For this task, we are going to use Filezilla, but you can also use scp or rsync from the command line.

1. Connect to **bigdatalab.polito.it** [host] on port 22, using the username and password you have been given.

2. The Filezilla interface is divided in two sections:
    a. on the left side you have the file system of your local machine (LABINF PC),
    b. on the right side the file system of the remote machine (the bigdatalab.polito.it gateway, with its own remote disk).

    Please note that you don't see the HDFS distributed file system in Filezilla.



Host: /bigdatalab.poli   Username: venturini   Password: ●●●●●●●   Port: 22

```
Status:     File transfer successful, transferred 4.1 KB in 1 second
Status:     Retrieving directory listing...
Command:    ls
Status:     Listing directory /home/dauin_user/venturini
Status:     Directory listing successful
Status:     Disconnected from server
```

Local site: /tmp/tempEserciziHadoop/Exercise1/target/

```
▼ Exercise1
    .settings
    ex1_data
  ▶ src
  ▶ target
▶ Exercise1WithCombiner
```

Filename	Filesize	Filetype	Last modified
..			
classes		Directory	03/09/2016 02:...
generated-s...		Directory	03/08/2016 03:...
maven-arch...		Directory	03/08/2016 03:...
maven-status		Directory	03/08/2016 03:...
test-classes		Directory	03/09/2016 02:...
Exercise1-1...	6.4 KB	Java Archive	03/09/2016 02:...

Remote site: /home/dauin_user/venturini

```
  baralis
  garza
  margara
  pulvirenti
▼ venturini
    .cache
```

Filename	Filesize	Filetype	Last modified	Permission:	Owner/Grou
..					
.cache		Directory	11/24/2015 ...	drwx------	venturini...
.bash...	88 B	File	02/19/2016 ...	-rw-------	venturini...
.bash...	220 B	File	04/09/2014	-rw-r--r--	venturini...
.bashrc	3.7 KB	File	04/09/2014	-rw-r--r--	venturini...
.profile	675 B	File	04/09/2014	-rw-r--r--	venturini...
.spar...	3 B	File	02/19/2016 ...	-rw-r--r--	venturini...
Exerc...	6.4 KB	Java Arc...	03/09/2016 ...	-rw-r--r--	venturini...

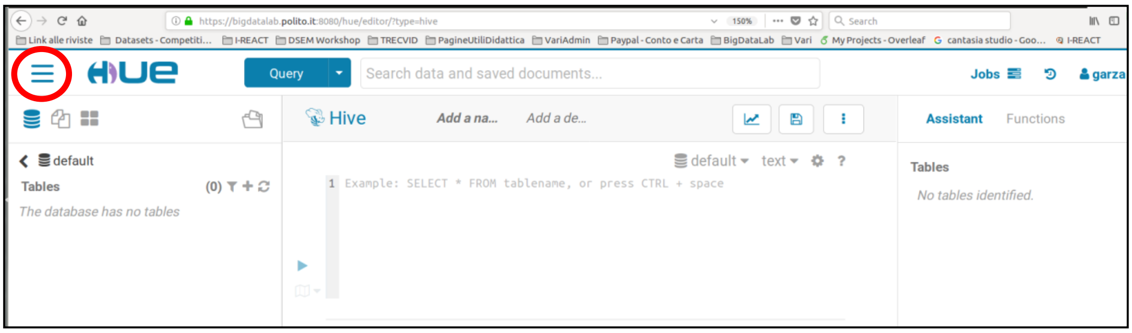
3. Locate on the local site your Exercise1-1.0.0.jar file (output of task 1), and a proper folder on the right (your home or a subfolder is fine). Right click on a file to find the upload command.

4. Check that the file was correctly transferred to the remote. You can either check the logs (above) or if the remote folder lists your file (on the right). In the figure, you see that a file, named Exercise1-1.0.0.jar, was copied to my home folder.

3. Manage HDFS through the HUE web interface

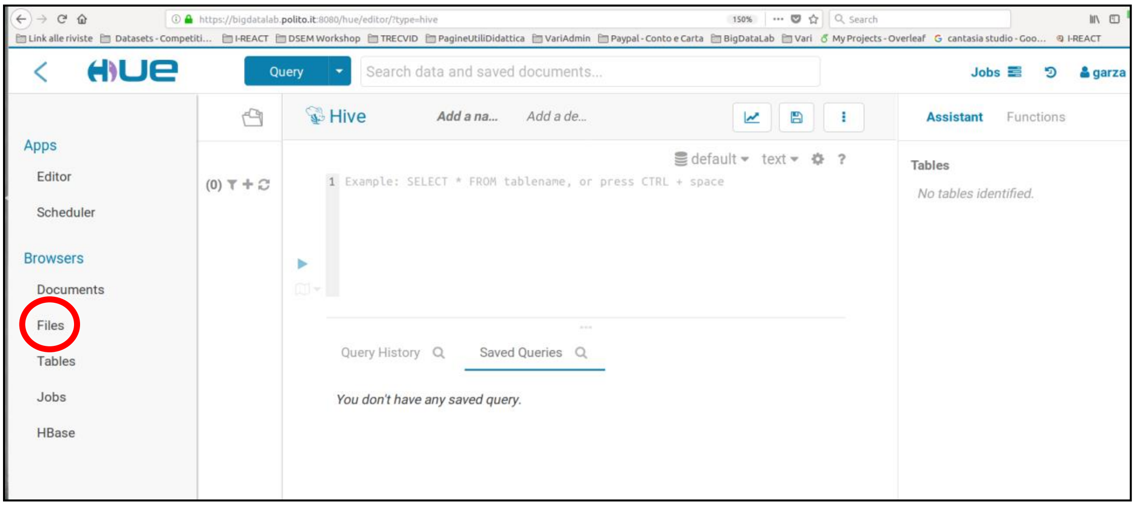
In this task, you will learn how to do basic management of the HDFS file system. To this goal, we will use a web interface called HUE. Again, you can do all the basic operations of this task on a command line; but this time, we advise you against this approach[1].

1. Go to https://bigdatalab.polito.it:8080 and login with your usual BigDataLab credentials.
2. Go to the “Browsers/Files” tab. You should find **your HDFS home**, as shown below. Note that this is not the same file system as in task 2 (i.e., the bigdatalab.polito.it physical disk), so you will find probably an empty folder now.[2]
   Your **HDFS home** is not located on the bigdatalab.polito.it machine, but is stored in the Hadoop cluster: the bigdatalab.polito.it is only a gateway that shows you the distributed HDFS contents.


https://bigdatalab.polito.it:8080/hue/editor/?type=hive
HUE
Query
Search data and saved documents...
Jobs
garza
Hive
Add a na...
Add a de...
Assistant
Functions
default
Tables
(0)
default
text
1 Example: SELECT * FROM tablename, or press CTRL + space
Tables

The database has no tables

No tables identified.



No tables identified.

You don't have any saved query.

3. Upload the sample file you have in ex1_data/document.txt, and check everything worked fine by opening it inside HUE. You should find two-three lines of sample text inside.

4. Find out on your own how to delete/move the files, or download them. It will be helpful in the next labs.

[1] You would need further configuration to access HDFS from a client on your local machine.

[2] If the difference between the two “homes” is not clear to you at this point, do not keep on. Spend some time to clearify your ideas.

4. Submit a job

Now we have everything we need to submit our sample application. It is finally time to open a command line. Using ssh, we will connect to the gateway we have used in task 2, where we should find the jar we copied at that time. From this gateway machine, we can (finally!) submit a job[3].

1. Connect, using SSH, on bigdatalab, with your usual BigDataLab credentials.

   ```
   >> ssh sXXXXXX@bigdatalab.polito.it
   ```

2. Now that you have a terminal on the gateway, launch a job using this command:

   ```
   >> hadoop jar Exercise1-1.0.0.jar
   it.polito.bigdata.hadoop.exercise1.DriverBigData 1
   your_input_file.txt ex1_out
   ```

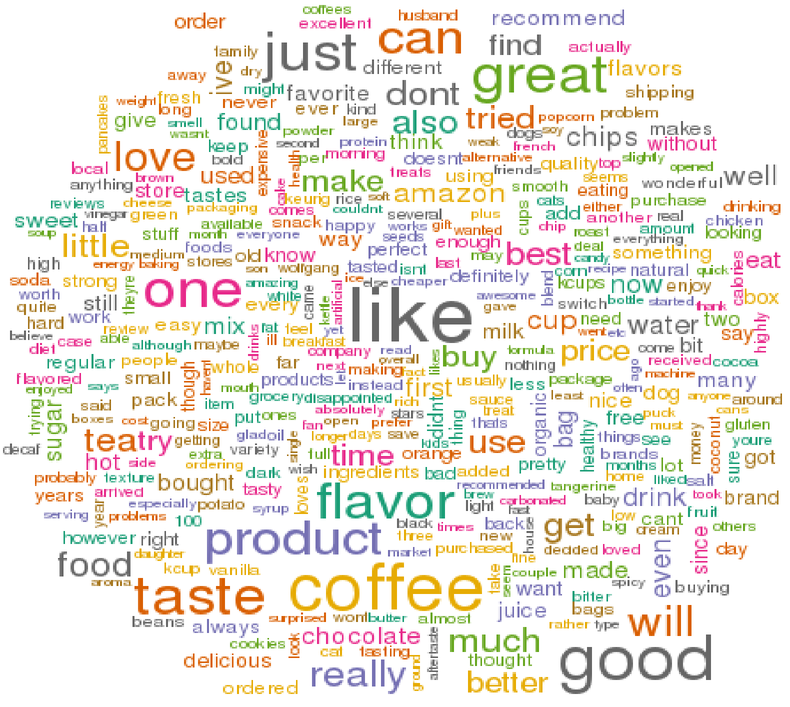
   where:
   - “**Exercise1-1.0.0.jar**” is the JAR file on the remote bigdata.polito.it disk (you can see it in Filezilla or in the terminal via SSH)
   - “**your_input_file.txt**” is the input file on the cluster HDFS (you can see the HDFS file system through the HUE web interface), a relative path starts in your home in HDFS, you can also use an absolute path in HDFS
   - “**ex1_out**” is the output folder in HDFS, not on the bigdatalab.polito.it disk, you can see its content in HUE (a relative path starts in your home in HDFS)
3. Find your job on HUE interface (JobBrowser), check its status (submitted, running, failed or succeeded) and find its stderr and stdout. Find also all the running jobs at the present time (i.e. not only yours).
4. Find the output file on HDFS (from HUE file browser) and see the results for the wordcount.
5. Try to re-run the same job. Does it succeed this time? What’s the problem?

[3] This gateway machine is simply the only machine in the cluster that can be accessed from outside, and the only one where users are allowed to launch jobs and manage HDFS.

5. Analyze the performances of your job

First, we need to setup some security configuration to access protected web interfaces on our cluster. First, we obtain a kerberos ticket that will grant us access to the system for a day; then, we will use Firefox to access the protected web interfaces (HUE is the only web UI you can access without completing this procedure).[4]
Keep in mind these operations for our next labs.

1. Open a new terminal on your local machine (do **not** use the previous terminal connected to bigdata.polito.it).
2. On the local terminal command line, type the command
   ```
   kinit sXXXXXX
   ```
   (then, you will be asked for your BigDataLab password)
3. Now you should have access to the full job history of YARN at https://ma1-bigdata.polito.it:19890/jobhistory/
4. How many mappers did your job instantiate? How many reducers?
5. Relaunch your job on a bigger file, that you can find on the cluster HDFS at the following path:
   /data/students/bigdata-01QYD/Lab1/finefoods_text.txt
   This a large collection of amazon reviews in the food category. Analyze the results. Can you understand any interesting facts from your results? Do you see any space for improvements in your analysis?
6. The following figure was done on a small sample of your data (10000 reviews). Is it consistent with what you found on the complete dataset? Do you think a small sample is enough to represent the whole?



[4] Details here: http://bigdata.polito.it/content/access-instructions

Bonus Task

A word count can be seen as a special case of an n-gram count, where n is equal to 1.
n-grams, in our context, are sets of contiguous words of length n.

For example, in the sentence “She sells seashells by the seashore“, 2-grams are: She sells, sells seashells, seashells by, by the, the seashore.

Modify your word count program to count 2-grams frequencies. Consider each line of text as a separate document, as in the amazon reviews file: so, do not count as contiguous words on separate lines.

What is the time/space complexity of your program? How long would you expect it to run, compared to the simple word count? Try to run it on the toy text and on the amazon reviews.