UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

Big Data Management (BDMA & MIRI Masters)
Session: HDFS (Training)

Alberto Abelló & Sergi Nadal

The purpose of this document is that you get familiar with the environment and tools required for the lab session. This is an optional task, thus no delivery is expected and no grade will be assigned. This document is far from being a fully-fledged tutorial, we refer you to the online manuals of HDFS, Avro and Parquet for in-depth guides.

Virtual machine

All tools you need are provided in the BDM virtual machine, accessible via *Virtech* or available to download. Please, refer to the provided manual for details on how to set up the virtual machine. To make the best out of the available time in class, please prepare your virtual machine in advance. Finally, notice that as lab sessions are performed in couples, only one virtual machine is available per team.

HDFS commands

To start HDFS in your virtual machine, run the following command:

```
/home/bdm/BDM_Software/hadoop/sbin/start-dfs.sh
```

To make sure that HDFS is correctly running, startup the browser and refer to `http://HOST:9870`. Notice that you must change the HOST, to the host IP assigned to your virtual machine. In the *Summary* section you should see that there is one live node. Now, note that if we try to explore a little bit on our HDFS directory and we run the next command we are going to obtain the error below since no user folder has been yet defined:

```
~/BDM_Software/hadoop/bin/hdfs dfs -ls
ls: '.': No such file or directory
```

In order to solve this, we write:

```
~/BDM_Software/hadoop/bin/hdfs dfs -mkdir /user
~/BDM_Software/hadoop/bin/hdfs dfs -mkdir /user/bdm
~/BDM_Software/hadoop/bin/hdfs dfs -chmod -R 777 /user/bdm/
```

In case you mess up and want to format HDFS (i.e., the NameNode) write the following command, and recreate the `/user/bdm` folder.

```
~/BDM_Software/hadoop/bin/hdfs namenode -format
```

Important note: In the case you do want to reformat your HDFS, delete the `data` directory before formating, to avoid inconsistencies.

Data generation

Download the enclosed Java project which can be imported as a *Maven Project* into IntelliJ on your lab machines (alternatively you can as well use Eclipse during the training period on your personal PCs). All executions can be run locally from IntelliJ just by replacing HOST within *Reader and

*Writer classes with IP assigned to your virtual machines. In the *Main* class (in package *bdm.labs.hdfs*) you can see that different arguments are expected, and will drive the behavior of the program. The simplest way to define such arguments is by creating a *Run Configuration* of type *Java Application* from the *Run* menu. In the tab menu called *Arguments* you will be able to edit such arguments, in each exercise you will be provided with the required arguments.
To be able to write from your local PC to the remote HDFS cluster, you will need to give specific WRITE permissions to your HDFS directory by executing the following command:

```
~/BDM_Software/hadoop/bin/hdfs dfs -chmod -R 777 /user/bdm
```

In this session we will use the *Adult* dataset[1], containing information about census and their income. You can check the files `adult.names` and `adult.avsc` (located in the *resources* directory of the Java project) to get a better understanding of the schema of data being used. The provided Java program will aid us on generating data with different formats. The run configuration arguments are the following (in the same order): *read/write format number_of_tuples path_for_file*. Precisely, we will explore the following formats (in bold the argument that will be used in the run configuration):

1. Plain text (`-plainText`).
2. SequenceFile (`-sequenceFile`).
3. Avro (`-avro`).
4. Parquet (`-parquet`).

First, generate a file with 10 milion rows as a plain text (i.e., with the running configuration `-write -plainText 10000000 adult.10milion.txt`) and upload it to HDFS with the following command: Try to explore a little bit more on what has been going on. You can check the file's content with the classic `cat` command for HDFS. **Beware that you will be printing 10 milion rows, better try this with smaller file sizes**.

```
~/BDM_Software/hadoop/bin/hdfs dfs -cat FILE_NAME
```

Alternatively, you can use the HDFS filesystem checking utility, as follows:

```
~/BDM_Software/hadoop/bin/hdfs fsck /user/bdm/FILE_NAME -files -blocks -locations
```

- **What is the size of the file in HDFS?**
- **How many blocks have been stored?**
- **What is the average block size? Discuss if such results make sense to you**

In order to delete files in HDFS with the following command:

```
~/BDM_Software/hadoop/bin/hdfs dfs -rm FILE_NAME
```

Writing and reading in different formats

Now we will explore different file formats. Generate files with 10000 rows as a plain text, and then as well with Sequence files, Avro, and Parquet formats, and insert them into HDFS. They correspond to the following run configurations:

```
-write -plainText 10000 adult.10000.txt
-write -sequenceFile 10000 adult.10000.seq
-write -avro 10000 adult.10000.avr
-write -parquet 10000 adult.10000.prq
```

[1] `https://archive.ics.uci.edu/ml/datasets/Adult`

Then, load these files into your HDFS using the put command. You can read each of your files, with the following standard commands (-cat):

```
~/BDM_Software/hadoop/bin/hdfs dfs -cat adult.10000.txt
~/BDM_Software/hadoop/bin/hdfs dfs -cat adult.10000.seq
~/BDM_Software/hadoop/bin/hdfs dfs -cat adult.10000.avr
~/BDM_Software/hadoop/bin/hdfs dfs -cat adult.10000.prq
```

- **What do your files look like now? Are they readable?**
- **What do you think is happening?**

Now, read the SequenceFile file again via the Java project:

```
-read -sequenceFile adult.10000.seq
```

- **Is your file readable now?**
- **What do you think the first column represents?**
- **Is it something given automatically or do we need to specify it?**
- **How is it built?**

Hint: Follow the code at the classes MyHDFSSequenceFileWriter *and* MyHDFSSequenceFileReader *for answers.*