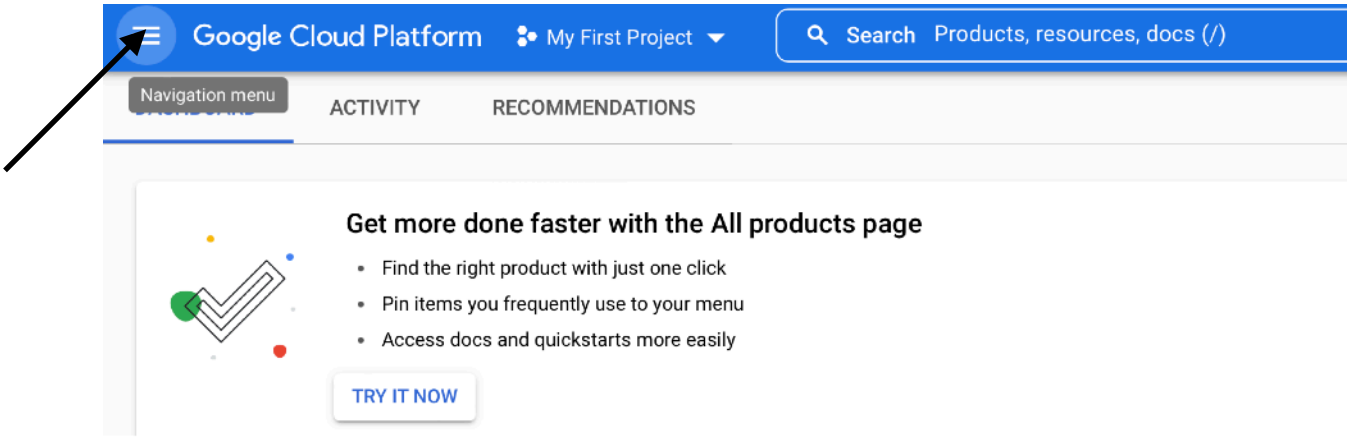
Big data processing on Google Cloud

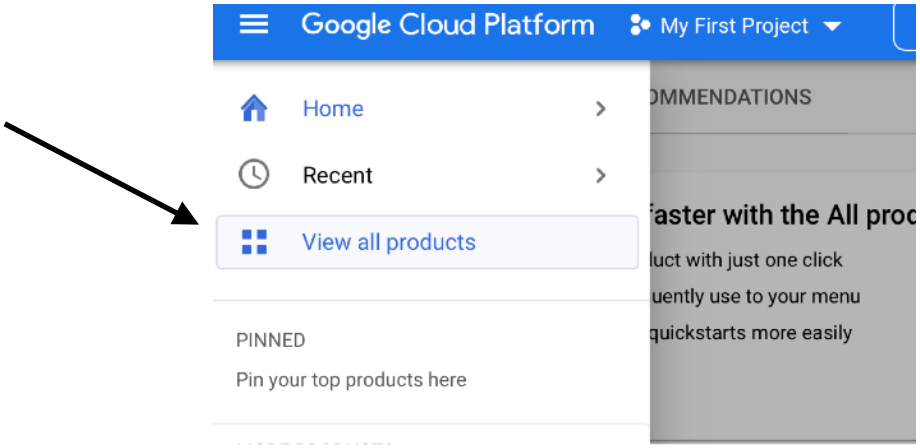
Creating and launching an Hadoop cluster, 1 Master and 2 Slaves

Please follow the steps illustrated by following images

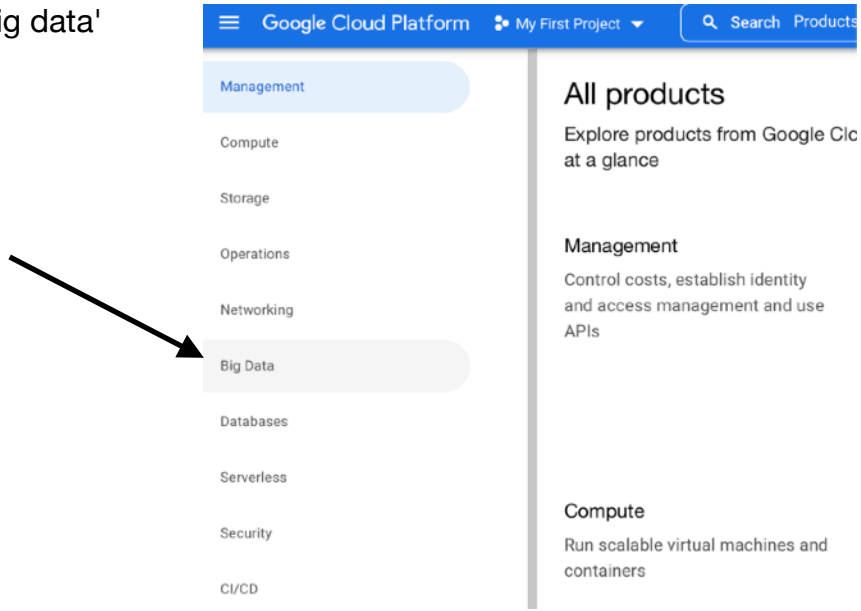
Starting from the initial home page (after having logged in) click on the Navigation menu



Then click on 'View all products'

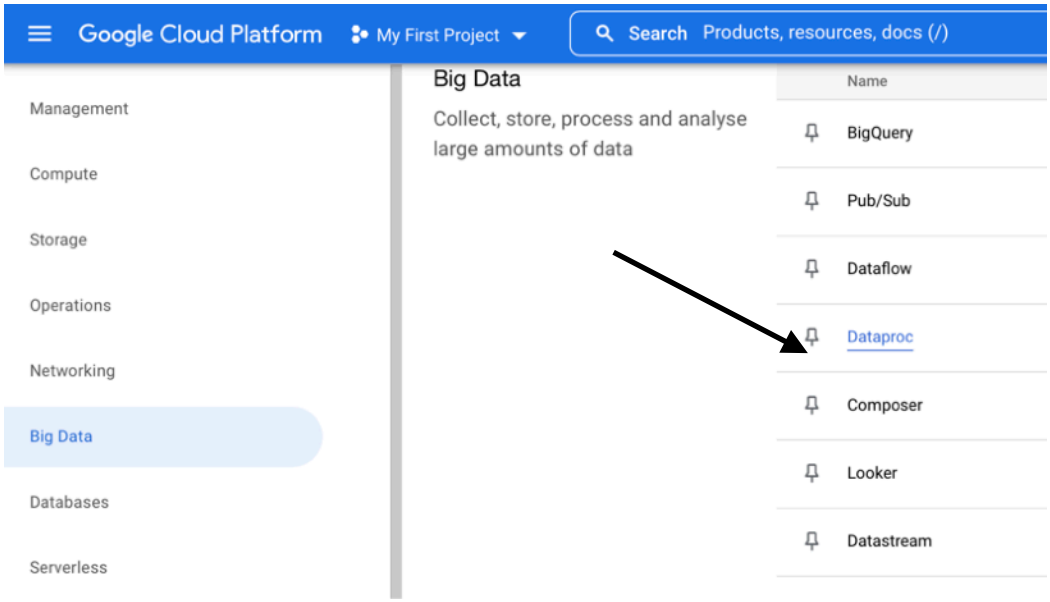


Now click on 'Big data'

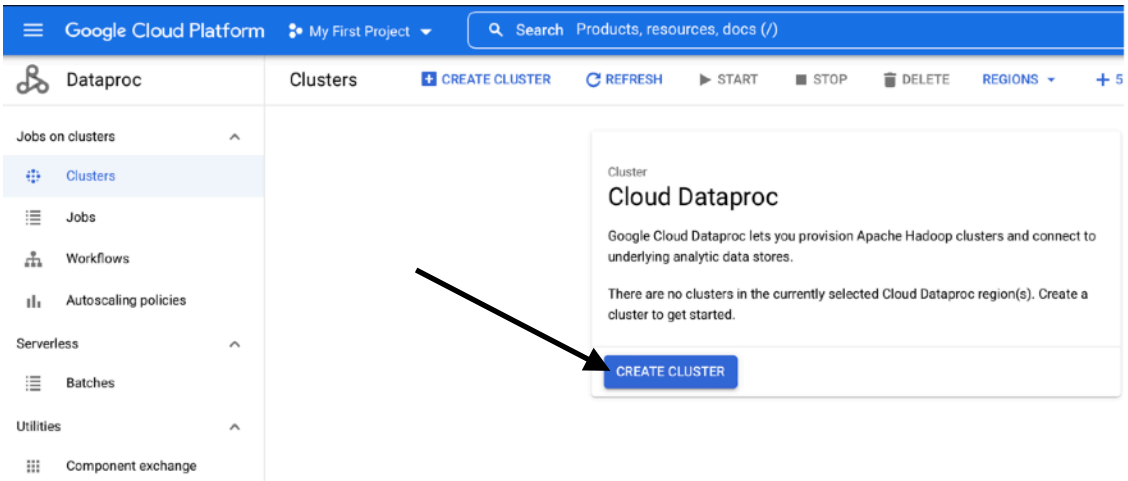


Big data processing on Google Cloud

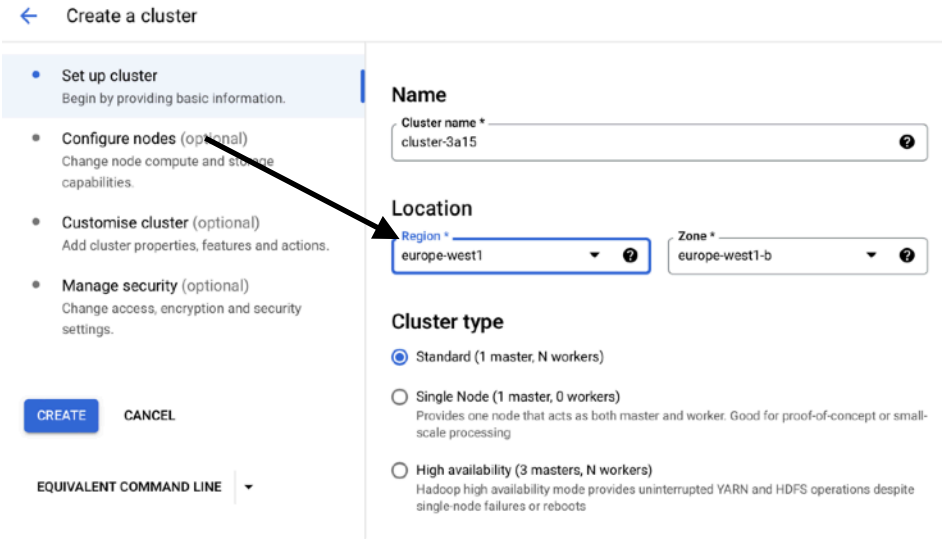
Then opt for Dataproc (which will allow for the creation of an Hadoop cluster).



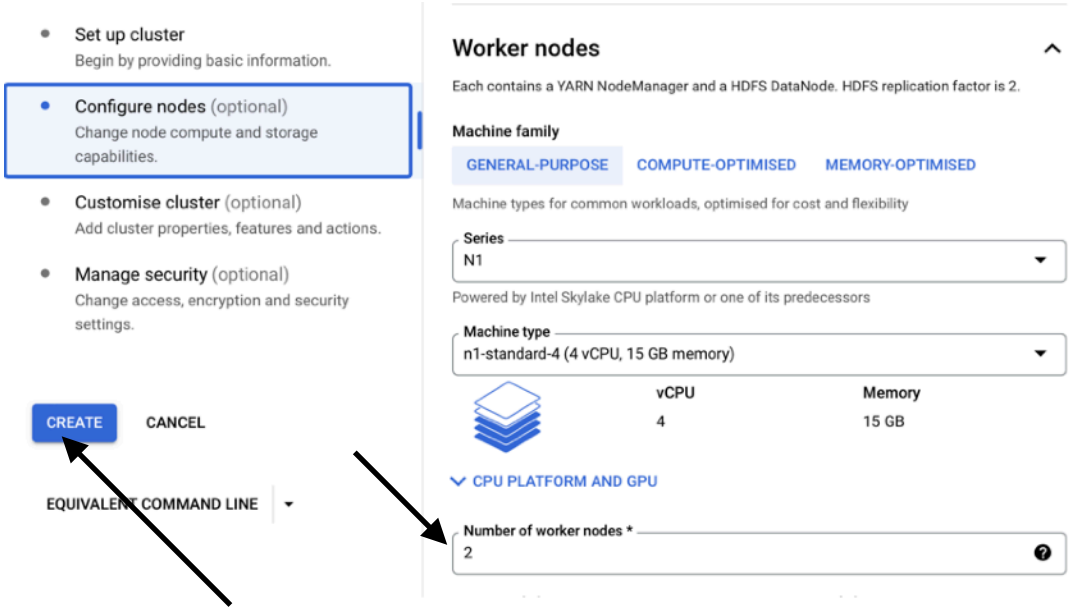
For cluster creation click on CREATE CLUSTER (opt for compute engine)



Now fill the regions like indicated below and leave the rest as pre-filled

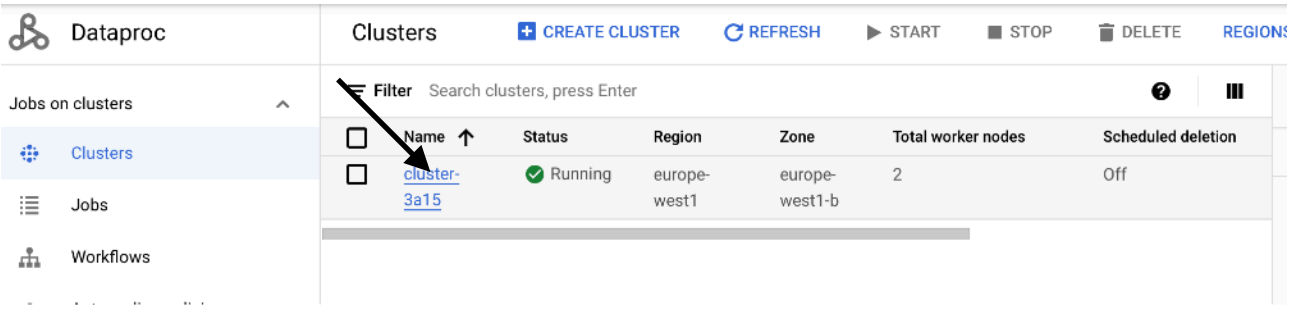


Concerning 'Configure node' you can observe that as a default setting you have 2 slaves (worker nodes). You can keep all these default settings.

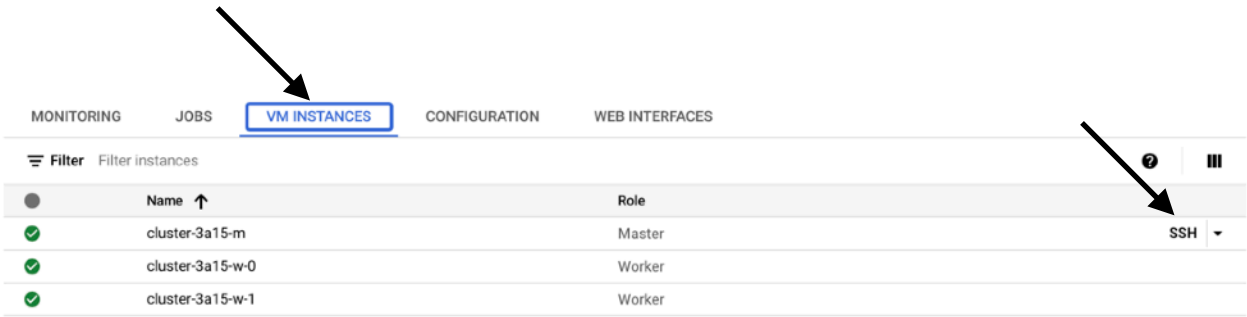


You can now CREATE the cluster (do not change other parameters)

Click now on the cluster link



Now click on the SSH button, this will enable an SSH connection with the Master



Big data processing on Google Cloud

You have now the possibility of using HDFS, MapReduce, Spark and Hive.

```
Connected, host fingerprint: ssh-rsa 0 D0:A5:35:F3:67:60:1C:00:A8:52:D0:5D:D7:EE
:20:9F:85:E3:C9:2D:AF:EA:9F:B4:15:33:82:B3:8E:38:41:14
Linux cluster-3a15-m 5.10.0-0.bpo.9-amd64 #1 SMP Debian 5.10.70-1~bpo10+1 (2021-
10-10) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
draclz@cluster-3a15-m:~$ ls
draclz@cluster-3a15-m:~$ pwd
/home/draclz
draclz@cluster-3a15-m:~$ hdfs dfs -ls /
Found 3 items
drwxrwxrwt   - hdfs hadoop          0 2022-02-01 13:31 /tmp
drwxrwxrwt   - hdfs hadoop          0 2022-02-01 13:31 /user
drwx-wx-wx   - hive hadoop          0 2022-02-01 13:31 /var
draclz@cluster-3a15-m:~$
```

For Spark in Python just use pyspark

```
draclz@cluster-3a15-m:~$ pyspark
Python 3.8.12 | packaged by conda-forge | (default, Oct 12 2021, 21:59:51)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newL
evel).
22/02/01 13:43:04 INFO org.apache.spark.SparkEnv: Registering MapOutputTracker
22/02/01 13:43:04 INFO org.apache.spark.SparkEnv: Registering BlockManagerMaster
22/02/01 13:43:04 INFO org.apache.spark.SparkEnv: Registering BlockManagerMasterHeartb
eat
22/02/01 13:43:04 INFO org.apache.spark.SparkEnv: Registering OutputCommitCoordinator
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 3.1.2
      /_/

Using Python version 3.8.12 (default, Oct 12 2021 21:59:51)
Spark context Web UI available at http://cluster-3a15-m.europe-west1-b.c.folkloric-gur
u-337910.internal:36787
Spark context available as 'sc' (master = yarn, app id = application_1643722259019_000
2).
SparkSession available as 'spark'.
>>> quit()
draclz@cluster-3a15-m:~$
```

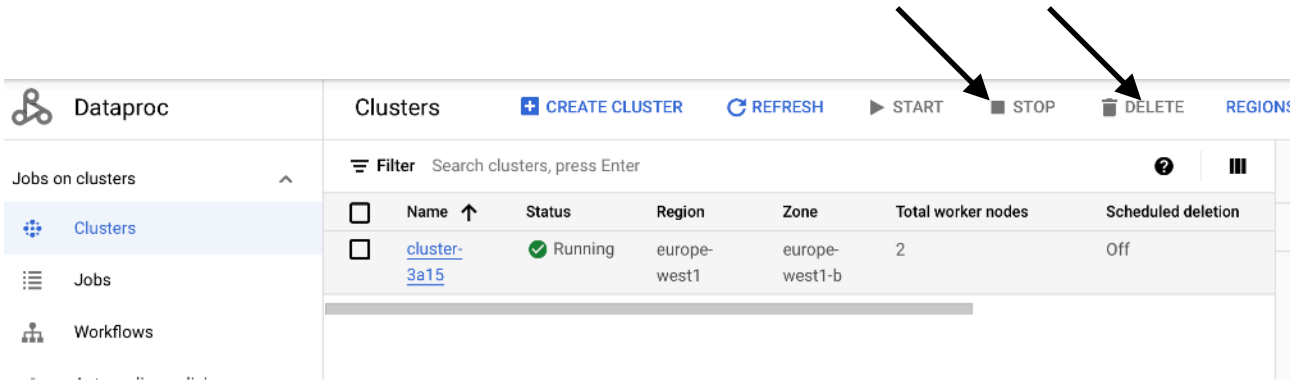
Concerning HDFS, with

```
hdfs dfsadmin -report
```

you get information about all living DataNodes (including their addresses).

IMPORTANT: once finished with your session, use the exit command for exiting the SSH terminal. -->Then go to the Google Cloud interface for the cluster VMs and stop the cluster

IMPORTANT: This will stop the cluster by stopping money consumption, just click on start to start again when needed. Note that there will be some minimal money consumption, so if you think that for several days/weeks you will not use the cluster, just DELETE it and re-CREATE when needed.



Run WordCount on the Google Cloud Cluster.

Once the SSH connection/terminal available (see previous pictures), proceed with the following steps

1. Download the mapper, reducer and a text file in you home directory, with the following commands

```
>wget https://www.dropbox.com/s/471k0286292ifho/mapper.py

> wget https://www.dropbox.com/s/e8s6f7rsiwts84m/reducer.py
```

Concerning the input text file:

```
> wget https://www.dropbox.com/s/hj8khqc94vsrzw9/shake.txt
```

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

```
chmod +x *.py
```

3. Now create a wc directory in the HDFS /user directory.

```
hdfs dfs -mkdir /user/wc
```

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

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

5. 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 shake.txt /user/wc/input/.
```

In order to check the status of shake.txt inside HDFS use

```
hdfs fsck /user/wc/input/shake.txt  -files -blocks -locations
```

In particular this will show number of blocks and respective DataNodes storing them.

6. You are ready now to launch our MapReduce job. Copy this command into the SSH cluster terminal. You will get several statistics upon job completion. The job results are in the 'output' HDFS directory. Each reduce task has produced its own file result.

```
hadoop jar /usr/lib/hadoop/hadoop-streaming-3.2.2.jar \
-input  /user/wc/input \
-output  /user/wc/output \
-file ~/mapper.py \
-mapper ~/mapper.py \
-file  ~/reducer.py \
-reducer ~/reducer.py
```

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

To see the output files (one per reduce task):

```
hdfs dfs -ls /user/wc/output
```

To see the content of these files:

```
hdfs dfs -cat /user/wc/output/*
```

If you want to use the combiner then use this variant (note that you re-use the reducer as a combiner, and that this holds for the word-count problem, but not necessarily for other problems, e.g. computing averages). Note that you have to use another output file.

```
hadoop jar /usr/lib/hadoop/hadoop-streaming-3.2.2.jar \
-input  /user/wc/input \
-output  /user/wc/output1 \
-file ~/mapper.py \
-mapper ~/mapper.py \
-file ~/reducer.py \
-reducer ~/reducer.py \
-combiner ~/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/hadoop-streaming-3.2.2.jar  \
-input  /user/wc/input \
-output  /user/wc/output2 \
-file ~/mapper.py \
-mapper ~/mapper.py \
-file  ~/reducer.py \
-reducer ~/reducer.py \
-jobconf mapred.reduce.tasks=3
```

7. 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.