Descargar unos ficheros de texto con la información del libro Moby Dick e indicar el resultado de ejecutar las siguientes instrucciones:

1. Recuperar el libro de Moby Dick del proyecto gutenberg
http://www.gutenberg.org/cache/epub/2701/pg2701.txt

2. Crear una carpeta ejercicioSpark en HDFS

$ start-dfs.sh
$ hdfs dfs -mkdir /ejerciciospark

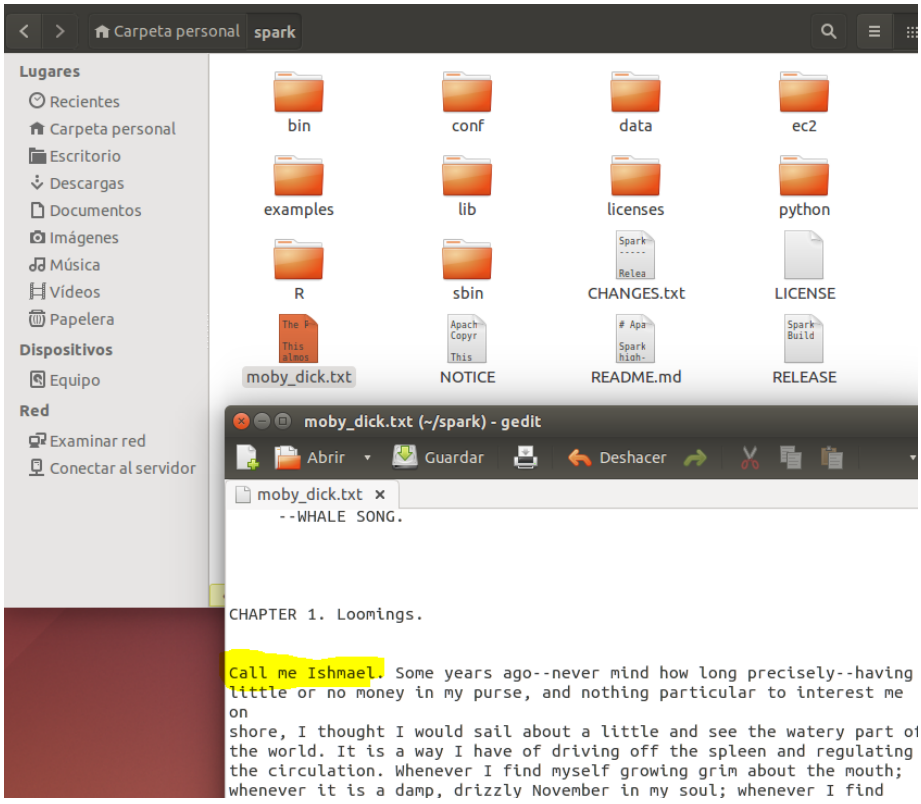
```
bigdata@bigdata:~$ start-dfs.sh
16/10/10 23:06:13 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
Starting namenodes on [localhost]
localhost: namenode running as process 3456. Stop it first.
localhost: datanode running as process 3587. Stop it first.
Starting secondary namenodes [0.0.0.0]
0.0.0.0: secondarynamenode running as process 3812. Stop it first.
16/10/10 23:06:21 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
bigdata@bigdata:~$ hdfs dfs -mkdir /ejerciciospark
16/10/10 23:06:30 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
mkdir: `/ejerciciospark': File exists
bigdata@bigdata:~$
```

3.

Subir el fichero a HDFS a la carpeta anterior

Para ello manualmente nos creamos una fichero en ubuntu en: equipo/home/bigdata/spark/

Y le copiamos el libro



El siguiente paso es copiarlo con este comando

$ hdfs dfs -put moby_dick.txt /ejerciciospark

```
bigdata@bigdata:~$ hdfs dfs -put moby_dick.txt /ejerciciospark
16/10/11 00:12:53 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
put: `moby_dick.txt': No such file or directory
bigdata@bigdata:~$
```

4. Comprobar que el libro está correctamente subido

Para ello nos situamos en la carpeta: home/bigdata/spark y ejecutamos el cat

cat moby_dick.txt

```
bigdata@bigdata:~/spark$ cat moby_dick.txt
```

Y vemos que:

```
bigdata@bigdata: ~/spark
bowsprit beneath, caught sight of the down-coming monster just as soon
as he.

"The whale, the whale! Up helm, up helm! Oh, all ye sweet powers of air,
now hug me close! Let not Starbuck die, if die he must, in a woman's
fainting fit. Up helm, I say--ye fools, the jaw! the jaw! Is this the
end of all my bursting prayers? all my life-long fidelities? Oh, Ahab,
Ahab, lo, thy work. Steady! helmsman, steady. Nay, nay! Up helm again!
He turns to meet us! Oh, his unappeasable brow drives on towards one,
whose duty tells him he cannot depart. My God, stand by me now!"
```

5. Acceder a pyspark

ejecutamos: pyspark

```
bigdata@bigdata:~/spark$ pyspark
```

```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 1.6.2
      /_/

Using Python version 2.7.6 (default, Mar 22 2014 22:59:56)
SparkContext available as sc, SQLContext available as sqlContext.
>>>
```

6. Indicar los comandos y el resultado de contar el número de líneas que tiene el fichero

Para hacer esta busqueda tenemos que declarar la variable mobydick, asignandole el fichero texto moby_dick.txt que hemos subido

textFile = sc.textFile("/ejercicioSpark/moby_dick.txt")

```
>>> textFile = sc.textFile("/ejercicioSpark/moby_dick.txt")
16/10/12 15:41:52 INFO storage.MemoryStore: Block broadcast_4 stored as values i
n memory (estimated size 191.9 KB, free 886.6 KB)
16/10/12 15:41:52 INFO storage.MemoryStore: Block broadcast_4_piece0 stored as b
ytes in memory (estimated size 22.1 KB, free 908.7 KB)
16/10/12 15:41:52 INFO storage.BlockManagerInfo: Added broadcast_4_piece0 in mem
ory on localhost:55396 (size: 22.1 KB, free: 517.3 MB)
16/10/12 15:41:52 INFO spark.SparkContext: Created broadcast 4 from textFile at
NativeMethodAccessorImpl.java:-2
>>>
```

Y ahora hacemos el conteo de todas las lineas que tienen un espacio enn la variable mobydick (nuestro texto)

textFile.filter(lambda line: " "in line).count()

o tambien: textFile.count()

7. Ejecutar un word count e indicar ordenadas alfabéticamente las dos últimas palabras que aparecen y el número de repeticiones

Debe ser algo así: wordCounts = textFile.flatMap(lambda line: line.split()).map(lambda word: (word, 1)).reduceByKey(lambda a, b: a+b)
pero …

8. Indicar las instrucciones y el valor devuelto del número de líneas en las que aparece la palabra Moby en el libro

Es como en el punto 7, pero en vez de contar líneas que tengan un espacio en blanco, un text.count() pues contamos las ineas que aparece la palabra moby

textFile.filter(lambda line: "Moby"in line).count()

9. Describe con tus palabras las diferencias de usar Spark frente a Map Reduce

Map reduce se creo para poder trabjar con enormes ficheros de datos, y spark es una evolucion posterior. El primero es de Apache y el segundo uun proyecto de la universidad de Berkeley.

Principalmente Mapreduce trabaja en disco y Spark en memoria. Por eso para hacer consultas de bases de datos se utiliza más spark y tiene más conexiones a distintas tipos de bases de datos.