

Memoria de ejercicios Hadoop

Tema HDFS

Ejercicio A – Canciones

1. Crear en local el directorio canciones

Se crea un directorio para almacenar los archivos .txt con la letra de las canciones. Y se comprueba que ha sido creado correctamente.

```
bigdata@bigdata:~/hadoop$ sudo mkdir canciones
bigdata@bigdata:~/hadoop$ ls
bin          ejercicios  LICENSE-binary  NOTICE.txt
canciones    etc         licenses-binary  README.txt
ejemploDir.txt  include    LICENSE.txt     sbin
EjemploMerge.txt lib         logs            share
ejemplo.txt    libexec    NOTICE-binary  wc-in
```

2. Crear los archivos con la letra de las canciones

Se crea un archivo .txt por cada canción con la letra correspondiente y se almacena en la carpeta canciones.

```
bigdata@bigdata:~/hadoop$ sudo nano ./canciones/nine_inch_nails.txt
```

```
GNU nano 4.8 ./canciones/nine_inch_nails
step right up march push
crawl right up on your knees
please greed feed (no time to hesitate)
I want a little bit I want a piece of it I think
he's losing it
I want to watch it come down
don't like the look of it don't like the taste
of it don't like the smell of it
I want to watch it come down
all the pigs are all lined up
I give you all that you want
take the skin and peel it back
now doesn't that make you feel better?
shove it up inside surprise! lies
stains like the blood on your teeth
bite chew suck away the tender parts
I want to break it up I want to smash it up
I want to fuck it up
I want to watch it come down
maybe afraid of it let's discredit it let's
pick away at it
I want to watch it come down
now doesn't that make you feel better?
the pigs have won tonight
now they can all sleep soundly
and everything is all right
[ 26 líneas escritas ]
^G Ver ayuda ^O Guardar ^W Buscar ^K Cortar Text ^J Justificar
^X Salir ^R Leer fich. ^\ Reemplazar ^U Pegar ^T Ortografía
```

```
bigdata@bigdata:~/hadoop$ sudo nano ./canciones/pink_floyd.txt
```

```
bigdata@bigdata:~/hadoop$ sudo nano ./canciones/dave_matthews_band.txt
```

3. Comprobación de los directorios que hay en HDFS en ese momento

Se comprueban los directorios albergados en HDFS para posteriormente crear otro con otro nombre.

```
bigdata@bigdata:~/hadoop$ hdfs dfs -du
0      0      dir1
29     29     dir4
138    138    dirEjemplo
59     59     ejemplosHDFS
0      0      ejemplosHDFS2
```

4. Creación de un nuevo directorio en HDFS

Se crea en HDFS un nuevo directorio para almacenar las canciones

```
bigdata@bigdata:~/hadoop$ hdfs dfs -mkdir /user/bigdata/canciones
```

5. Subida de archivos

Se suben todos los archivos .txt con las canciones al directorio en HDFS y posteriormente se comprueba que se han subido correctamente.

```
bigdata@bigdata:~/hadoop$ hdfs dfs -put ./canciones/pink_floyd.txt ./canciones/nine_inch_nails.txt ./canciones/dave_matthews_band.txt /user/bigdata/canciones/
```

```
bigdata@bigdata:~/hadoop$ hdfs dfs -ls ./canciones
Found 3 items
-rw-r--r--  1 bigdata supergroup      2184 2022-10-28 11:42 canciones/dave_matthews_band.txt
-rw-r--r--  1 bigdata supergroup       839 2022-10-28 11:42 canciones/nine_inch_nails.txt
-rw-r--r--  1 bigdata supergroup      1108 2022-10-28 11:42 canciones/pink_floyd.txt
```

6. Ejecución del programa

Una vez subido todos los archivos con las canciones se ejecuta sobre estos archivos el programa de contador de palabras, y se indica un directorio como output para los resultados.

```
bigdata@bigdata:~/hadoop$ hadoop jar $HADOOP_HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.3.1.jar wordcount /user/bigdata/canciones /user/bigdata/canciones-out
2022-10-28 12:20:18,099 INFO client.DefaultHARMFalloverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2022-10-28 12:20:18,649 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/bigdata/.staging/job_1666947893127_0002
2022-10-28 12:20:19,129 INFO input.FileInputFormat: Total input files to process : 3
2022-10-28 12:20:19,232 INFO mapreduce.JobSubmitter: number of splits:3
2022-10-28 12:20:19,491 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1666947893127_0002
2022-10-28 12:20:19,492 INFO mapreduce.JobSubmitter: Executing with tokens: []
2022-10-28 12:20:19,851 INFO conf.Configuration: resource-types.xml not found
2022-10-28 12:20:19,853 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2022-10-28 12:20:19,989 INFO impl.YarnClientImpl: Submitted application application_1666947893127_0002
2022-10-28 12:20:20,098 INFO mapreduce.Job: The url to track the job: http://bigdata:8088/proxy/application_1666947893127_0002/
2022-10-28 12:20:20,102 INFO mapreduce.Job: Running job: job_1666947893127_0002
2022-10-28 12:20:32,721 INFO mapreduce.Job: Job job_1666947893127_0002 running in uber mode : false
2022-10-28 12:20:32,722 INFO mapreduce.Job: map 0% reduce 0%
2022-10-28 12:20:49,252 INFO mapreduce.Job: map 67% reduce 0%
2022-10-28 12:20:50,272 INFO mapreduce.Job: map 100% reduce 0%
2022-10-28 12:20:56,322 INFO mapreduce.Job: map 100% reduce 100%
2022-10-28 12:20:56,376 INFO mapreduce.Job: Job job_1666947893127_0002 completed successfully
2022-10-28 12:20:56,566 INFO mapreduce.Job: Counters: 55
File System Counters
FILE: Number of bytes read=4589
FILE: Number of bytes written=1099651
```

7. Solución del programa

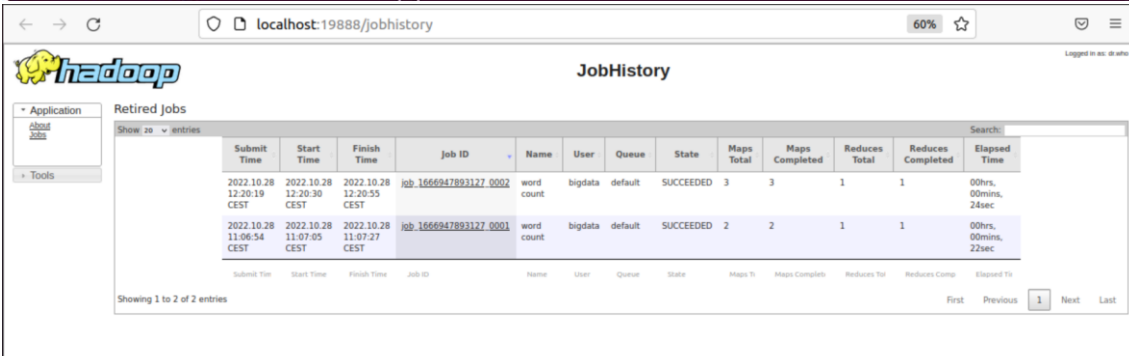
Se comprueba la salida del Word-count sacando por pantalla todas las palabras y número de veces repetidas entre las tres canciones.

```
bigdata@bigdata:~/hadoop$ hdfs dfs -cat /user/bigdata/canciones-out/*
"keep      1
(no        1
.....!.....!.....!.....!      1
All        1
Almost     2
And         7
Big         1
Bus         1
But         9
Come        2
Deep        1
Don't       9
Drinking    1
Dry         2
Everything   1
From        1
Greedy      1
Head        1
Here        1
```

8. Respuesta a las preguntas

- Job History: se lanza el demonio por separado, se comprueba que se ha levantado correctamente y se accede a la <Direccion>:<Puerto> correspondiente.

```
data@bigdata:~/hadoop$ mr-jobhistory-daemon.sh start historyserve
NING: Use of this script to start the MR JobHistory daemon is deprecated.
NING: Attempting to execute replacement "mapred --daemon start" instead.
data@bigdata:~/hadoop$ jps
78 Jps
8 DataNode
5 SecondaryNameNode
4 ResourceManager
1 NameNode
23 JobHistoryServer
2 NodeManager
```



The screenshot shows the Hadoop JobHistory web interface in a browser. The URL is localhost:19888/jobhistory. The page title is "JobHistory". On the left, there is a sidebar with "Application" (cloud icon) and "Tools". The main content area is titled "Retired Jobs" and shows a table with 20 entries. The table has columns: Submit Time, Start Time, Finish Time, Job ID, Name, User, Queue, State, Maps Total, Maps Completed, Reduces Total, Reduces Completed, and Elapsed Time. Two jobs are visible in the table, both with a state of "SUCCEEDED".

Submit Time	Start Time	Finish Time	Job ID	Name	User	Queue	State	Maps Total	Maps Completed	Reduces Total	Reduces Completed	Elapsed Time
2022.10.28 12:20:19 CEST	2022.10.28 12:20:30 CEST	2022.10.28 12:20:55 CEST	job_1666947893127_0002	word count	bigdata	default	SUCCEEDED	3	3	1	1	00hrs, 00mins, 24sec
2022.10.28 11:06:54 CEST	2022.10.28 11:07:05 CEST	2022.10.28 11:07:27 CEST	job_1666947893127_0001	word count	bigdata	default	SUCCEEDED	2	2	1	1	00hrs, 00mins, 22sec

Showing 1 to 2 of 2 entries

Una vez dentro del jobhistory se puede observar que el estado de la ejecución realizada es "Succeeded", la variable Map total es 3.

- b. Palabra más repetida: "the" -> 33 veces.

```
the      33
```

- c. Palabras repetidas 6 veces: "we", "our", "charade", "laugh" y "You". Se excluye "ha" ya que realmente ha dividido el resultado en "ha" y "ha,".

```
bigdata@bigdata:~/hadoop$ hdfs dfs -cat /user/bigdata/canciones-out/* | grep "6"
You      6
charade  6
ha       6
ha,      6
laugh    6
our      6
we       6
```

- d. Palabra "eyes": Se repite 4 veces. Misma casuística que con la palabra "ha".

```
bigdata@bigdata:~/hadoop$ hdfs dfs -cat /user/bigdata/canciones-out/* | grep "eyes"
eyes     3
eyes,    1
```

Ejercicio B – Sudoku

1. Crear en local el directorio

Se crea un directorio para almacenar los archivos .txt con los tableros de los sudokus a solucionar. Y comprobar que se creó correctamente.

```
bigdata@bigdata:~/hadoop$ mkdir ./sudoku
bigdata@bigdata:~/hadoop$ ls
bin          ejercicios  LICENSE-binary  NOTICE.txt  wc -ln
canciones   etc        licenses-binary README.txt
ejemploDir.txt include    LICENSE.txt    sbin
EjemploMerge.txt lib        logs           share
ejemplo.txt libexec    NOTICE-binary sudoku
```

2. Crear el primer archivo

Se crea un archivo para almacenar el primer ejercicio a solucionar.

```
GNU nano 4.8
? 4 6 3 ? ? 8 2 5
? ? ? ? ? 2 4 ? ?
8 2 ? ? 6 ? ? 7 ?
7 ? ? 4 ? ? ? ? 2
3 8 ? ? ? ? 6 ? 9
? 6 ? 2 8 ? ? ? ?
? 7 ? ? ? 5 ? 3 ?
5 3 8 ? ? 7 ? ? ?
? 9 ? ? ? ? ? ? 6
```


3. Ejecutar programa

Se ejecuta el programa de solución de un sudoku con el fichero anteriormente creado como entrada. Observamos la solución por pantalla.

```
bigdata@bigdata:~/hadoop$ hadoop jar $HADOOP_HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.3.1.jar sudoku ./sudoku/sudoku1.txt

Solving ./sudoku/sudoku1.txt
1 4 6 3 7 9 8 2 5
9 5 7 8 1 2 4 6 3
8 2 3 5 6 4 9 7 1
7 1 5 4 9 6 3 8 2
3 8 2 7 5 1 6 4 9
4 6 9 2 8 3 5 1 7
6 7 1 9 4 5 2 3 8
5 3 8 6 2 7 1 9 4
2 9 4 1 3 8 7 5 6

Found 1 solutions
```

4. Crear fichero, segundo ejercicio

Se crea el fichero con el tablero del segundo sudoku para posteriormente ejecutar el programa de solución.

```
GNU nano 4.8      ./sudoku/sudoku2.txt
? 6 8 ? 2 14 15 ? ? 5 ? 16 10 ? ? ?
? 5 ? 12 ? ? 9 ? 10 ? 3 ? 7 ? ? ?
9 15 ? ? 8 ? ? 12 ? 2 ? ? ? 13 ? 5
13 ? ? 4 ? 6 ? ? 7 ? ? ? ? ? ? ?
2 10 ? 14 5 4 ? 11 ? ? ? 13 15 ? ? ?
? 8 11 ? 14 ? ? ? ? ? ? 2 ? 3 ? ?
? ? 13 3 ? ? 10 ? ? ? ? ? ? ? 2 12
? ? ? ? ? ? ? 6 ? 9 4 ? ? ? ? 8
12 ? ? 8 11 15 ? ? 3 ? ? 5 ? ? ? 2
? 11 15 ? ? 9 12 ? ? ? 13 6 ? ? ? 14
7 ? ? ? ? 2 ? ? ? ? ? 14 12 ? ? 13
10 ? 2 ? ? ? ? ? 4 15 ? ? ? ? ? 7
? ? 7 ? 12 13 ? 5 8 ? ? 9 ? 14 15 ?
? ? 6 15 ? ? ? ? ? ? ? ? ? ? ? ?
? 14 ? ? ? ? ? ? ? 10 ? 3 6 ? 4
3 ? 5 ? ? ? 8 10 13 ? 7 ? 9 12 ? 11
```

5. Segunda ejecución programa

Se ejecuta por segunda vez el programa de solución de sudoku, pero esta vez la entrada es el segundo tablero creado. Se observa por pantalla el resultado y se responden las preguntas.

```
bigdata@bigdata:~/hadoop$ hadoop jar $HADOOP_HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.3.1.jar sudoku ./sudoku/sudoku2.txt

Solving ./sudoku/sudoku2.txt
1 6 8 7 2 14 15 13 9 5 12 16 10 11 4 3
14 5 16 12 1 11 9 4 10 13 3 8 7 2 6 15
9 15 3 11 8 10 7 12 6 2 14 4 1 13 16 5
13 2 10 4 3 6 5 16 7 1 15 11 14 8 12 9
2 10 12 14 5 4 1 11 16 3 8 13 15 9 7 6
4 8 11 9 14 12 16 7 15 6 5 2 13 3 10 1
6 16 13 3 9 8 10 15 14 11 1 7 4 5 2 12
15 7 1 5 13 3 2 6 12 9 4 10 11 16 14 8
12 13 14 8 11 15 4 1 3 7 16 5 6 10 9 2
5 11 15 1 7 9 12 8 2 10 13 6 16 4 3 14
7 9 4 16 10 2 6 3 1 8 11 14 12 15 5 13
10 3 2 6 16 5 13 14 4 15 9 12 8 1 11 7
11 1 7 10 12 13 3 5 8 4 6 9 2 14 15 16
8 12 6 15 4 1 14 9 11 16 2 3 5 7 13 10
16 14 9 13 15 7 11 2 5 12 10 1 3 6 8 4
3 4 5 2 6 16 8 10 13 14 7 15 9 12 1 11

Found 1 solutions
```

Para responder las preguntas he tenido en cuenta que la primera posición es la “1” y que el primer número son las columnas y el segundo las filas.

- 1,8: 15
- 7,5: 11
- 9,10: 7
- 13,6: 13

Tema Map Reduce

Ejercicio Diccionario

1. Crear directorio de trabajo

Se crea en local la carpeta “diccionario” para almacenar todos los archivos del ejercicio, y se comprueba que se ha creado correctamente.

```
bigdata@bigdata:~/hadoop$ mkdir ./diccionario
bigdata@bigdata:~/hadoop$ ls
bin          ejercicios   licenses-binary  sbin
canciones    etc          LICENSE.txt      share
diccionario  include     logs             sudoku
ejemploDir.txt  lib         NOTICE-binary   wc-ln
EjemploMerge.txt libexec      NOTICE.txt
ejemplo.txt    LICENSE-binary README.txt
bigdata@bigdata:~/hadoop$
```

2. Descargar archivos

Descargar los archivos con las traducciones, se almacena en un archivo diferente para cada uno de los idiomas.

```
bigdata@bigdata:~/hadoop$ ls ./diccionario/
French.txt  German.txt  Italian.txt  Spanish.txt
```

3. Merge Idiomas

Se unen cada uno de los archivos con las traducciones en un único archivo, ya que Hadoop funciona mejor con un único archivo más grande.

```
bigdata@bigdata:~/hadoop/diccionario$ cat French.txt German.txt Italian.txt Spanish.txt > Dictionary.txt
```

4. Subir diccionario a HDFS

Se crea un nuevo directorio en HDFS, se sube el archivo con nombre “diccionario.txt”, ya que en el código Java se ha especificado ese nombre de fichero como entrada, por último, se comprueba que se haya subido correctamente.

```
bigdata@bigdata:~/hadoop/diccionario$ hdfs dfs -mkdir /user/bigdata/mapreduce
```

```
bigdata@bigdata:~/hadoop/diccionario$ hdfs dfs -put Dictionary.txt /user/bigdata/mapreduce/diccionario.txt
```

```
bigdata@bigdata:~/hadoop/diccionario$ hdfs dfs -ls /user/bigdata/mapreduce
Found 1 items
-rw-r--r--  1 bigdata supergroup      725372 2022-10-31 10:57 /user/bigdata/mapreduce/diccionario.txt
```

5. Crear el archivo.java

Se crea un nuevo archivo con el código Java expuesto en el PDF.

```
GNU nano 4.8 Dictionary.java
import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.KeyValueTextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class Dictionary{

    public static class WordMapper extends Mapper <Text, Text, Text, Text> {

        private Text word = new Text();

        public void map (Text key, Text value, Context context) throws IOException, InterruptedException {
            StringTokenizer itr = new StringTokenizer(value.toString());
            while(itr.hasMoreTokens()){
                word.set(itr.nextToken());
                context.write(key, word);
            }
        }
    }
}
```


6. Compilar el fichero

Una vez guardado en un archivo.java el código del programa, este se compila con el objetivo de comprobar la correcta definición del código.

```
bigdata@bigdata:~/hadoop/diccionario$ sudo javac -classpath $HADOOP_HOME/share/hadoop/common/hadoop-common-3.3.1.jar:$HADOOP_HOME/share/hadoop/common/lib/hadoop-annotations-3.3.1.jar:$HADOOP_HOME/share/hadoop/mapreduce/hadoop-mapreduce-client-core-3.3.1.jar /home/bigdata/hadoop/diccionario/Dictionary.java
```

7. Crear el .jar

Se crea el archivo ejecutable con el código y clases definidas anteriormente en el .java. Se comprueba que se ha creado correctamente.

```
bigdata@bigdata:~/hadoop/diccionario$ sudo jar cf dict.jar Dictionary*.class
bigdata@bigdata:~/hadoop/diccionario$ ls
'Dictionary$AllTranslationsReducer.class'  dict.jar
'Dictionary$WordMapper.class'             French.txt
Dictionary.class                          German.txt
Dictionary.java                           Italian.txt
Dictionary.txt                             Spanish.txt
```

8. Ejecución del programa

Se ejecuta el proceso map reduce utilizando el ejecutable.jar

```
bigdata@bigdata:~/hadoop/diccionario$ hadoop jar dict.jar Dictionary
2022-10-31 11:49:44,382 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2022-10-31 11:49:44,948 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
2022-10-31 11:49:45,003 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/bigdata/.staging/job_1666947893127_0006
2022-10-31 11:49:45,462 INFO input.FileInputFormat: Total input files to process : 1
2022-10-31 11:49:45,622 INFO mapreduce.JobSubmitter: number of splits:1
2022-10-31 11:49:46,063 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1666947893127_0006
2022-10-31 11:49:46,063 INFO mapreduce.JobSubmitter: Executing with tokens: []
2022-10-31 11:49:46,425 INFO conf.Configuration: resource-types.xml not found
```

9. Devolver la traducción de pig

Se realiza una búsqueda de la palabra Pig y se saca por pantalla la traducción en los diferentes idiomas de esta.

```
bigdata@bigdata:~/hadoop/diccionario$ hdfs dfs -cat /user/bigdata/output/part-r-000000 | grep -w pig
pig |Schwein (n)|el puerco|Schwein (n)|el chancho[Noun]
```


Tema Hive

Ejercicio Deuda 2020

Lo primero que hago es acceder a la URL y descargar el archivo. Una vez descargado lo abro y modifico la cabecera, las ultimas dos filas y el formato de las celdas de deuda. Quedaría de la siguiente manera:

	A	B	C	D	E	F	G	H
13	2020	01	ANDALUCIA	04	ALMERIA	001	Abia	293
14	2020	01	ANDALUCIA	04	ALMERIA	002	Abucena	0
15	2020	01	ANDALUCIA	04	ALMERIA	003	Adra	14602
16	2020	01	ANDALUCIA	04	ALMERIA	004	Albánchez	0
17	2020	01	ANDALUCIA	04	ALMERIA	005	Alboloduy	0
18	2020	01	ANDALUCIA	04	ALMERIA	006	Albox	18584
19	2020	01	ANDALUCIA	04	ALMERIA	007	Alcolea	0
20	2020	01	ANDALUCIA	04	ALMERIA	008	Alcóntar	57
21	2020	01	ANDALUCIA	04	ALMERIA	009	Alcudia de Monteaquid	11
22	2020	01	ANDALUCIA	04	ALMERIA	010	Alhabia	1827
23	2020	01	ANDALUCIA	04	ALMERIA	011	Alhama de Almería	1793
24	2020	01	ANDALUCIA	04	ALMERIA	012	Alcún	86
25	2020	01	ANDALUCIA	04	ALMERIA	013	Almería	74761
26	2020	01	ANDALUCIA	04	ALMERIA	014	Almócita	2
27	2020	01	ANDALUCIA	04	ALMERIA	015	Alsodux	0
28	2020	01	ANDALUCIA	04	ALMERIA	016	Antas	626
29	2020	01	ANDALUCIA	04	ALMERIA	017	Arboleas	304
30	2020	01	ANDALUCIA	04	ALMERIA	018	Armuña de Almanzora	0
31	2020	01	ANDALUCIA	04	ALMERIA	019	Baza	50

Dejo el año, ya que en el siguiente apartado pide crear una tabla con el primer campo siendo año. Lo guardo como .csv y cierro el archivo.

Entro en hive y empiezo con la ejecución de los comandos, lo primero es introducir el comando de "CREATE TABLE", con los parámetros adecuados:

```
hive> CREATE TABLE deuda_2020(año INT, cod_comunidad STRING, comuni-  
dad STRING, cod_prov STRING, provincia  
> STRING, cod_municipio STRING, municipio STRING, deuda INT) ROW  
FORMAT DELIMITED FIELDS  
> TERMINATED BY '\;';
```

Cargo el fichero que he descargado y modificado anteriormente en la tabla que acabo de crear (deuda_2020).

```
hive> LOAD DATA LOCAL INPATH '/home/bigdata/ejemplosHive/Deuda2020.c  
sv' INTO TABLE deuda_2020;
```

Compruebo que se han cargado correctamente los datos.

```
hive> SELECT deuda_2020.* FROM deuda_2020;
```

```

2020 17 C.VALENCIANA 46 VALENCIA 904 Benicull de Xúquer
0
2020 18 CEUTA 51 CEUTA 001 Ceuta 178530
2020 19 MELILLA 52 MELILLA 001 Melilla 76628
Time taken: 0.391 seconds, Fetched: 8143 row(s)

```

Con la siguiente consulta obtengo el municipio con la tercera deuda más grande. Para ello, selecciono el municipio y deuda, ordenado descendientemente, me salto los dos primeros registros y saco por pantalla solo el siguiente a esos dos saltados, es decir, el tercero.

```

hive> SELECT municipio, deuda FROM deuda_2020 ORDER BY deuda DESC LI
MIT 1 OFFSET 2;

```

```

Total MapReduce CPU Time Spent: 4 seconds 360 msec
OK
Jerez de la Frontera
799843
Time taken: 29.636 seconds, Fetched: 1 row(s)

```

Con la siguiente consulta obtengo el municipio con más deuda de Andalucía, para ello selecciono comunidad, municipio y deuda, siempre y cuando la comunidad sea Andalucía y los ordeno descendientemente, sacando por pantalla solo el primer registro de la lista.

```

hive> SELECT comunidad, municipio, deuda from deuda_2020 WHERE comun
idad = "ANDALUCIA" ORDER BY deuda DESC LIMIT 1;

```

```

Total MapReduce CPU Time Spent: 5 seconds 620 msec
OK
ANDALUCIA Jerez de la Frontera
799843

```

Ejercicios Opcionales

Selecciono la comunidad y la suma de la deuda de cada municipio agrupando por comunidad, y lo ordeno por comunidad.

```

hive> SELECT comunidad, SUM(deuda) FROM deuda_2020 GROUP BY comunidad ORDER BY comunidad;

```

```

ANDALUCIA 5134185
ARAGON 788845
ASTURIAS 190392
C.VALENCIANA 1299980
CANARIAS 135005
CANTABRIA 54705
CASTILLA-LEON 791022
CASTILLA-MANCHA 504112
CATALUÑA 2973799
CEUTA 178530
EXTREMADURA 159504
GALICIA 216430
ILLES BALEARS 233714
MADRID 3651525
MELILLA 76628
MURCIA 761201
NAVARRA 102052
PAIS VASCO 374903
RIOJA 52476

```

Selecciono la suma de la deuda que tienen los municipios que empiezan por 'A', 'E', 'I', 'O' o 'U'.

```
hive> SELECT SUM(deuda)
> FROM deuda_2020
> WHERE (municipio LIKE 'A%' OR municipio LIKE 'E%'
> OR municipio LIKE 'I%' OR municipio LIKE 'O%' OR
> municipio LIKE 'U%')
> ;
```

```
Total MapReduce CPU Time Spent: 4 seconds 670 msec
OK
2184407
Time taken: 31.459 seconds, Fetched: 1 row(s)
```

Para la última parte, primero compruebo que la consulta funcione, y posteriormente extraigo el archivo a un directorio local. Para la consulta selecciono la comunidad, el municipio y la deuda, de todos los municipios de Cataluña los cuales tengan una deuda superior a 10.000.

```
hive> SELECT comunidad, municipio, deuda
> FROM deuda_2020
> WHERE comunidad = 'CATALUÑA' AND deuda > 10000
> ORDER BY deuda DESC;
```

```
Total MapReduce CPU Time Spent: 4 seconds 790 msec
OK
CATALUÑA      Barcelona      801480
CATALUÑA      Lleida         134259
CATALUÑA      Tarragona     119088
CATALUÑA      Terrassa     101715
CATALUÑA      Mataró        97411
CATALUÑA      Reus          93069
CATALUÑA      Sabadell      76444
CATALUÑA      Hospitalet de Llobregat (L') 71056
CATALUÑA      Sant Cugat del Vallès 56241
CATALUÑA      Girona        49453
CATALUÑA      Vilanova i la Geltrú 47935
CATALUÑA      Cornellà de Llobregat 41382
```

```
CATALUÑA      Castell-Platja d'Aro 20053
CATALUÑA      Montcada i Reixac 16925
CATALUÑA      Castelldefels 16336
CATALUÑA      Vilassar de Mar 16082
CATALUÑA      Martorell     15932
CATALUÑA      Gavà          15361
CATALUÑA      Amposta       15293
CATALUÑA      Valls         15043
CATALUÑA      Olot          14716
CATALUÑA      Cambrils     14631
CATALUÑA      Molins de Rei 13987
CATALUÑA      Sant Feliu de Llobregat 13939
CATALUÑA      Esplugues de Llobregat 13770
CATALUÑA      Caldes de Montbui 13750
CATALUÑA      Franqueses del Vallès (Les) 13631
CATALUÑA      Ametlla de Mar (L') 12564
CATALUÑA      Viladecans    11796
CATALUÑA      Vilafranca del Penedès 11594
CATALUÑA      Sant Joan Despi 10401
CATALUÑA      Mollerussa    10166
Time taken: 29.932 seconds, Fetched: 44 row(s)
```

Una vez que compruebo que la salida es adecuada, extraigo el fichero.

```
hive> INSERT OVERWRITE LOCAL DIRECTORY '/home/bigdata/ejemplosHive/R
RESULTADO/'
> ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
> SELECT comunidad, municipio, deuda
> FROM deuda_2020
> WHERE comunidad = 'CATALUÑA' AND deuda > 10000
> ORDER BY deuda DESC;
```



```
OK
Time taken: 32.412 seconds
```

Adjunto el fichero en este documento:



Discografía Pink Floyd

Lo primero, es crear el directorio donde voy a guardar el fichero y posteriormente, crear dicho fichero e introducir la información de los discos. Guardo y salgo del mismo.

```
bigdata@bigdata:~$ sudo mkdir /home/bigdata/ejemplosHive/discografiaPinkFloyd/
bigdata@bigdata:~$ sudo nano /home/bigdata/ejemplosHive/discografiaPinkFloyd/discografia.txt
```

```
...data/ejemplosHive/discografiaPinkFloyd/discografia.txt Modificado
1967,The Piper at the Gates of Dawn,131,6
1968,A Saucerful of Secrets,999,9
1969,Music from the Film More,153,9
1969,Ummagumma,74,5
1970,Atom Heart Mother,55,1
1972,Obscured by Clouds,46,6
1973,The Dark Side of the Moon,1,1
1975,Wish you Were Here,1,1
1977,Animals,3,2
1979,The Wall,1,3
1983,The Final Cut,6,1
1987,A Momentary Lapse of Reason,3,3
1994,The Division Bell,1,1
2014,The Endless River,3,1
```

Creo la tabla con las cuatro columnas que tenemos, año de lanzamiento, nombre del disco, ranking alcanzado en EEUU y Reino Unido.

```
bigdata@bigdata:~$ hive -e "CREATE TABLE discografia (anio INT, nombre_disco STRING, ranking EEUU INT, ranking UK INT) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';"
```

```
Hive Session ID = 02743307-99ff-4da6-8bb5-80908590186b
OK
Time taken: 1.862 seconds
```

Cargo el fichero anteriormente creado y modificado en la tabla “discografía”.

```
bigdata@bigdata:~$ hive -e "LOAD DATA LOCAL INPATH '/home/bigdata/ejemplosHive/discografiaPinkFloyd/discografia.txt' INTO TABLE discografia;"
```

```
Loading data to table default.discografia
OK
Time taken: 3.15 seconds
```

Entro a Hive y compruebo que los datos se han cargado correctamente ejecutando un "SELECT" para toda la tabla.

```
bigdata@bigdata:~$ hive
```

```
hive> SELECT * FROM discografia;
OK
1967    The Piper at the Gates of Dawn    131    6
1968    A Saucerful of Secrets    999    9
1969    Music from the Film More    153    9
1969    Ummagumma    74    5
1970    Atom Heart Mother    55    1
1972    Obscured by Clouds    46    6
1973    The Dark Side of the Moon    1    1
1975    Wish you Were Here    1    1
1977    Animals    3    2
1979    The Wall    1    3
1983    The Final Cut    6    1
1987    A Momentary Lapse of Reason    3    3
1994    The Division Bell    1    1
2014    The Endless River    3    1
Time taken: 0.286 seconds, Fetched: 14 row(s)
```

Posteriormente, empiezo con las consultas pedidas en el resto del ejercicio.

Selecciono el año y nombre del disco los cuales llegaron a ser del 1 al 5 más escuchados, ordeno por año para que sea más fácil de comprender la salida.

```
hive> SELECT anio, nombre_disco
> FROM discografia
> WHERE ranking_EEUU <= 5 AND ranking_UK <= 5
> ORDER BY anio DESC;
```

```
Total MapReduce CPU Time Spent: 5 seconds 510 msec
OK
2014    The Endless River
1994    The Division Bell
1987    A Momentary Lapse of Reason
1979    The Wall
1977    Animals
1975    Wish you Were Here
1973    The Dark Side of the Moon
Time taken: 36.614 seconds, Fetched: 7 row(s)
```

Selecciono la posición máxima (peor) y mínima (mejor) que obtuvieron los discos de Pink Floyd. Primero, realizo dicha consulta sobre el ranking de EEUU.

```
hive> SELECT MAX(ranking_EEUU), MIN(ranking_EEUU)
> FROM discografia;
```

```
Total MapReduce CPU Time Spent: 4 seconds 390 msec
OK
999      1
```

Por último, realizo la misma consulta pero sobre el ranking de Reino Unido.

```
hive> SELECT MAX(ranking_UK), MIN(ranking_UK)
> FROM discografia;
```

```
Total MapReduce CPU Time Spent: 3 seconds 730 msec
OK
9      1
```

TEMA Hbase

Discografía Pink Floyd

Siguiendo las instrucciones del ejercicio, lo primero que debemos hacer es crear un namespace en HBase con el nombre de discografía.

```
hbase:001:0> create_namespace 'discografia'
Took 0.7790 seconds
```

Una vez creado el namespace discografía, debemos definir y crear una tabla con solo dos columnas.

```
hbase:003:0> create 'discografia:albums', 'nombre_disco', 'rankin
g'
Created table discografia:albums
Took 0.6953 seconds
```

La tabla la he definido con el nombre 'albums'. Ahora, para comprobar que se ha creado correctamente hacemos un describe.


```

hbase:004:0> describe 'discografía:albums'
Table discografía:albums is ENABLED
discografía:albums
COLUMN FAMILIES DESCRIPTION
{NAME => 'nombre_disco', BLOOMFILTER => 'ROW', IN_MEMORY => 'false', VERSIONS => '1', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', COMPRESSION => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}

{NAME => 'ranking', BLOOMFILTER => 'ROW', IN_MEMORY => 'false', VERSIONS => '1', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', COMPRESSION => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}

2 row(s)
Quota is disabled
Took 0.2079 seconds

```

Vemos que efectivamente se ha creado con las dos columnas necesarias. Posteriormente, cargamos cada disco con su información relativa en la base de datos. Pongo captura de los primeros registros almacenados, y los comandos utilizados para ello. He utilizado como clave el formato <Año de lanzamiento>#<Número de disco lanzado ese año>:

```

hbase:006:0> put 'discografía:albums', '1967#1', 'nombre_disco', 'The Piper at the Gates of Dawn'
Took 0.1104 seconds
hbase:007:0> put 'discografía:albums', '1967#1', 'ranking:EEUU', 131
Took 0.0101 seconds
hbase:008:0> put 'discografía:albums', '1967#1', 'ranking:UK', 6
Took 0.0044 seconds
hbase:009:0> scan 'discografía:albums'
ROW COLUMN+CELL
1967#1 column=nombre_disco:, timestamp=2022-11-16T09:16:49.167, value=The Piper at the Gates of Dawn
1967#1 column=ranking:EEUU, timestamp=2022-11-16T09:17:15.250, value=131
1967#1 column=ranking:UK, timestamp=2022-11-16T09:17:27.631, value=6
1 row(s)
Took 0.0718 seconds

```

Una vez comprobado que las primeras inserciones han ido bien, he preparado un archivo “canciones.txt” con los comandos para almacenar directamente toda la discografía.

```
GNU nano 4.8      canciones.txt
put 'discografía:albums', '1968#1', 'nombre_disco', 'A Saucerful>
put 'discografía:albums', '1968#1', 'ranking:EEUU', '-'
put 'discografía:albums', '1968#1', 'ranking:UK', 9

put 'discografía:albums', '1969#1', 'nombre_disco', 'Music from >
put 'discografía:albums', '1969#1', 'ranking:EEUU', 153
put 'discografía:albums', '1969#1', 'ranking:UK', 9

put 'discografía:albums', '1969#2', 'nombre_disco', 'Ummagumma (>
put 'discografía:albums', '1969#2', 'ranking:EEUU', 74
put 'discografía:albums', '1969#2', 'ranking:UK', 5

put 'discografía:albums', '1970#1', 'nombre_disco', 'Atom Heart >
put 'discografía:albums', '1970#1', 'ranking:EEUU', 55
put 'discografía:albums', '1970#1', 'ranking:UK', 1

put 'discografía:albums', '1971#1', 'nombre_disco', 'Meddle'
put 'discografía:albums', '1971#1', 'ranking:EEUU', 70
put 'discografía:albums', '1971#1', 'ranking:UK', 3

put 'discografía:albums', '1972#1', 'nombre_disco', 'Obscured by>
put 'discografía:albums', '1972#1', 'ranking:EEUU', 46
put 'discografía:albums', '1972#1', 'ranking:UK', 6

put 'discografía:albums', '1973#1', 'nombre_disco', 'The Dark Si>
put 'discografía:albums', '1973#1', 'ranking:EEUU', 1

[ 57 líneas escritas ]
^G Ver ayuda  ^O Guardar  ^W Buscar  ^K Cortar Tex ^J Justificar
^X Salir      ^R Leer fich.^_ Reemplazar ^U Pegar     ^T Ortografía
```

Ejecuto el archivo para que se carguen todos los datos

```
bigdata@bigdata:~/hbase$ ./bin/hbase shell /home/bigdata/ejemplos
HBase/canciones.txt
```

Resultado ejecución:

```
Took 0.0067 seconds
Took 0.0067 seconds
Took 0.0119 seconds
Took 0.0056 seconds
Took 0.0165 seconds
Took 0.0104 seconds
Took 0.0150 seconds
Took 0.0051 seconds
Took 0.0040 seconds
Took 0.0086 seconds

HBase Shell
Use "help" to get list of supported commands.
Use "exit" to quit this interactive shell.
For Reference, please visit: http://hbase.apache.org/2.0/book.html#shell
Version 2.4.8, rf844d09157d9dce6c54fcd53975b7a45865ee9ac, Wed Oct 27 08:48:57 PDT 2021
Took 0.0035 seconds
```

Comprobación del correcto almacenamiento:

```

hbase:001:0> scan 'discografia:albums'
ROW                                COLUMN+CELL
1967#1                             column=nombre_disco:, timestamp=2022-11-16T09:16:49.167, value=The Piper at the Gates of Dawn
1967#1                             column=ranking:EEUU, timestamp=2022-11-16T09:17:15.250, value=131
1967#1                             column=ranking:UK, timestamp=2022-11-16T09:17:27.631, value=6
1968#1                             column=nombre_disco:, timestamp=2022-11-16T09:39:44.998, value=A Saucerful of Secrets
1968#1                             column=ranking:EEUU, timestamp=2022-11-16T09:39:45.101, value=-
1968#1                             column=ranking:UK, timestamp=2022-11-16T09:39:45.150, value=9
1969#1                             column=nombre_disco:, timestamp=2022-11-16T09:39:45.212, value=Music from the Film More
1969#1                             column=ranking:EEUU, timestamp=2022-11-16T09:39:45.242, value=153
1969#1                             column=ranking:UK, timestamp=2022-11-16T09:39:45.270, value=9
1969#2                             column=nombre_disco:, timestamp=2022-11-16T09:39:45.305, value=Ummagumma (Disco 2)
1969#2                             column=ranking:EEUU, timestamp=2022-11-16T09:39:45.322, value=74
1969#2                             column=ranking:UK, timestamp=2022-11-16T09:39:45.347, value=5
1970#1                             column=nombre_disco:, timestamp=2022-11-16T09:39:45.390, value=Atom Heart Mother
1970#1                             column=ranking:EEUU, timestamp=2022-11-16T09:39:45.419, value=55
1970#1                             column=ranking:UK, timestamp=2022-11-16T09:39:45.502, value=1
1971#1                             column=nombre_disco:, timestamp=2022-11-16T09:39:45.535, value=Meddle
1971#1                             column=ranking:EEUU, timestamp=2022-11-16T09:39:45.555, value=70
1971#1                             column=ranking:UK, timestamp=2022-11-16T09:39:45.625, value=3
1972#1                             column=nombre_disco:, timestamp=2022-11-16T09:39:45.647, value=Obscured by Clouds
1972#1                             column=ranking:EEUU, timestamp=2022-11-16T09:39:45.709, value=46
1972#1                             column=ranking:UK, timestamp=2022-11-16T09:39:45.721, value=6
1973#1                             column=nombre_disco:, timestamp=2022-11-16T09:39:45.745, value=The Dark Side of the Moon
1973#1                             column=ranking:EEUU, timestamp=2022-11-16T09:39:45.768, value=1
1973#1                             column=ranking:UK, timestamp=2022-11-16T09:39:45.789, value=1

```

Por último, expongo la captura de la consulta pedida en el apartado final (álbum del año 1968).

```

hbase:002:0> get 'discografia:albums', '1968#1'
COLUMN                                CELL
nombre_disco:                         timestamp=2022-11-16T09:39:44.998, value=A Saucerful of Secrets
ranking:EEUU                         timestamp=2022-11-16T09:39:45.101, value=-
ranking:UK                           timestamp=2022-11-16T09:39:45.150, value=9
1 row(s)
Took 0.0762 seconds

```