Ejercicios Sesión 03 HDFS: *FileFormats* & Compresión

Big Data Management

25 de abril de 2020

1. FileFormat

1.1. Ejercicio 1

Escribir y leer en distintos formatos:

- plainText
- sequenceFile
- avro
- parquet

Recuerde que para generar ficheros una vez que tenga el programa ejecutable *HDFS-1.0.jar* es usando el comando:

```
java -jar HDFS-0.1.jar write -fileformat #registros fichero
```

Y para deserializar(si es necesario) y leer el fichero de forma legible, es usando el comando:

```
java -jar HDFS-0.1.jar read -fileformat fichero
```

También le recordamos que para insertar ficheros en HDFS tiene diversas posibilidades ya vistas anteriormente:

```
hadoop-2.10.0/bin/hdfs dfs -D dfs.blocksize=Xm -D dfs.replication=Y -
    copyFromLocal ficheroOrigen destino
hadoop-2.10.0/bin/hdfs dfs -D dfs.blocksize=Xm -D dfs.replication=Y -put
    fichero
```

1. Crea ficheros de los cuatro formatos implementados con 10.000 registros e insertalos en HDFS.
2. Lea los ficheros insertados en HDFS con *hadoop-2.10.0/bin/hdfs dfs -cat*, son legibles? por qué?

3. Ahora use el programa *HDFS-1.0.jar* para leer los ficheros no legibles, lo son ahora? por qué?

4. Considere el fichero en formato SequenceFile, qué representa la primera columna? cómo se ha construido?. Investigue el código fuente si considera necesario.

1.2. Ejercicio 2

A continuación hagamos comparativas en espacio y tiempo lectura.

1. Genere un fichero de cata tipo de formato fijando a 1 el número de registros a crear.
2. Cuál es el tamaño de cada fichero?
3. Calcule la relación de tamaños entre los distintos formatos y *-plainText*
4. Por qué de dichas relaciones, explique considerando la estructura de cada formato
5. Ahora considere los ficheros creados en el Ejercicio 1, de 10.000 registros cada uno y conteste a las mismas cuestiones

   a) Cuál es el tamaño de cada fichero?

   b) Calcule la relación de tamaños entre los distintos formatos y *-plainText*

   c) Por qué de dichas relaciones, explique considerando la estructura de cada formato

6. Calcula el tiempo que se tarda en leer cada fichero de 10.000 registros usando *HDFS-1.0.jar* y compárelo con su tamaño. A qué se deben esas diferencias en tiempos de lectura?

	-plainText	SequenceFile	Avro	Parquet
tamaño				
tiempo lectura				

2. Compresión

Como se ha visto en la teoría, existen diferentes tipos de compresión y que les invito a investigar. Acudan al código fuente, en el *package org.adilazh1.hdfs.writer* y en cada clase, en el método *open()*, tenemos qué compresión se está usando.

2.1. Ejercicio 1

En este ejercicio nos quedamos únicamente en la compresión de un fichero con formato *SequenceFile*. En la clase *MyHDFSSequenceFileWriter* vemos que tenemos definido *SequenceFile.Writer.compression(CompressionType value)* donde CompressionType puede ser NONE, RECORD, BLOCK y por otra parte tenemos CompressionCodec sin definir y por tanto es el Codec por defecto[1] y qué podemos escoger *B2zip* o *Gzip*.

1. Modifique el código fuente para definir CompressionType para los tres tipos posibles, compile el programa y crea ficheros con cada tipo de compresión con 2 millones de registros. Inserte dichos ficheros en HDFS
2. Qué hace cada tipo de Compresión: NONE, RECORD, BLOCK?
3. Complete la tabla y discute los resultados

Nivel	tiempo inserción	tiempo lectura	#bloques	tamaño
NONE				
RECORD				
BLOCK				

[1] https://hadoop.apache.org/docs/r2.10.0/api/org/apache/hadoop/io/compress/CompressionCodec.html

Conclusiones Generales?