SCALA

Lectura de ficheros planos en el siguiente ejemplo utilizamos count

```
val quijote = sc.textFile("dbfs:/FileStore/tables/el_quijote.txt")//guardamos el fichero del quijote en una variable del mismo nombre

quijote.count//  utilizamos count
```

▸ (1) Spark Jobs

```
quijote: org.apache.spark.rdd.RDD[String] = dbfs:/FileStore/tables/el_quijote.txt MapPartitionsRDD[53] at textFile at command-3618742328255304:1
res36: Long = 2186

Command took 0.75 seconds -- by josemanuel.cerrato@bosonit.com at 3/5/2022, 10:53:09 on My Cluster
```

lectura de una tablas aplicando tipo de formato y otras opciones
utilizamos display para que muestre la tabla en el formato correcto

```
//aplicar opciones de lectura de archivo
val mnm_dataFrames = spark.read.format("csv")//formato de lectura
                    .option("inferSchema",true)//true da como resultado ejm(tx=string, 20 integer, etc) en caso de falso todo se combierte a stringType
                    .option("header",true)//muestra el encabesado del fichero, en caso de false = c1, c2, c3 que estan por defecto de cabecera
                    .option("sep",",")//cual es el delimitador
                    .load("dbfs:/FileStore/tables/mnm_dataset.csv")//direccion de archivo a utilizar
display(mnm_dataFrames)
```

Scala

▸ (3) Spark Jobs

▸ mnm_dataFrames: org.apache.spark.sql.DataFrame = [State: string, Color: string ... 1 more field]

	State	Color	Count
1	TX	Red	20
2	NV	Blue	66
3	CO	Blue	79
4	OR	Blue	71
5	WA	Yellow	93
6	WY	Blue	16
7	CA	Yellow	53

Truncated results, showing first 1000 rows.

Command took 3.40 seconds -- by josemanuel.cerrato@bosonit.com at 4/5/2022, 9:26:39 on My Cluster

Cmd 4

lectura mostrando numero x numero de lineas

```
//show = muestra el contenido del marco de datos en una tabla o fichero
val quijote = sc.textFile("dbfs:/FileStore/tables/el_quijote.txt")

//El método toDF() proporciona una forma muy concisa de crear un marco de datos
val numLinias = quijote.toDF()
//metodo show con numero de filas en este caso mostrara 5 filas del dataframes
numLinias.show(5)
```

▸ (1) Spark Jobs

▸ numLinias: org.apache.spark.sql.DataFrame = [value: string]

```
+--------------------+
|               value|
+--------------------+
|DON QUIJOTE DE LA...|
|Miguel de Cervant...|
|                    |
|       PRIMERA PARTE|
|CAPÍTULO 1: Que ...|
+--------------------+
only showing top 5 rows
```

podemos utilizar truncate en la lectura de ficheros planos por defecto mostrara las 20 primeras linias

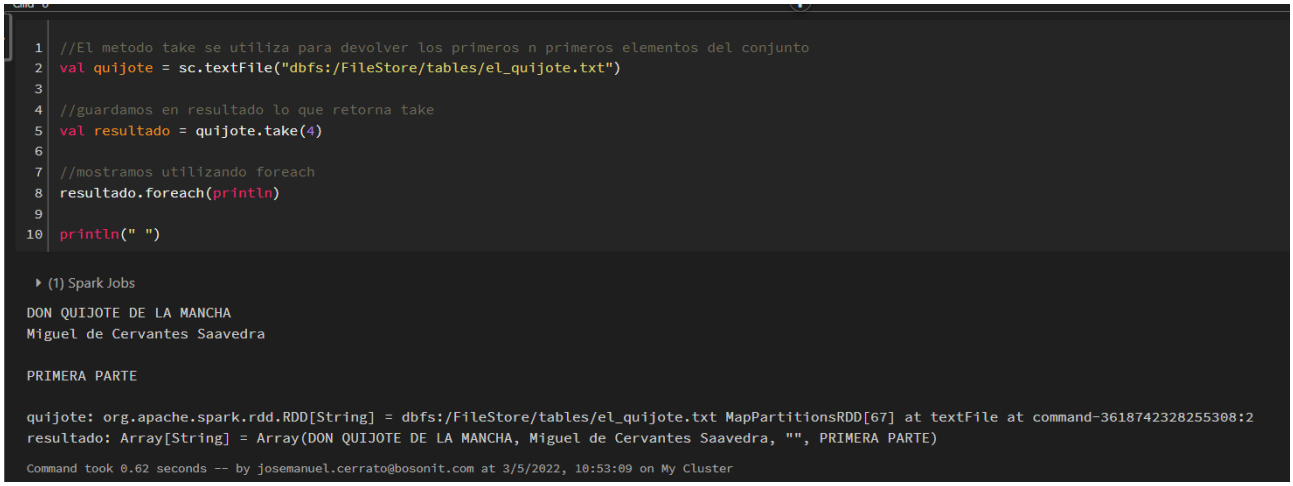
```
1  //show = muestra el contenido del marco de datos en una tabla o fichero
2  val quijote = sc.textFile("dbfs:/FileStore/tables/el_quijote.txt")
3
4  //El método toDF() proporciona una forma muy concisa de crear un marco de datos
5  val numLinias = quijote.toDF()
6  //metodo show truncate por defecto mostrara 20 filas
7  numLinias.show(truncate = true)
```

```
▸ (1) Spark Jobs
▸ numLinias:  org.apache.spark.sql.DataFrame = [value: string]
+--------------------+
|               value|
+--------------------+
|DON QUIJOTE DE LA...|
|Miguel de Cervant...|
|                    |
|       PRIMERA PARTE|
|CAPÍTULO 1: Que ...|
|En un lugar de la...|
|Tuvo muchas veces...|
|En resolución, e...|
|historia más cie...|
|Decía él, que e...|
|En efecto, remata...|
|Imaginábase el p...|
|linaje y patria, ...|
|Limpias, pues, su...|
```

El método **take()** pertenece al miembro de valor de la clase List . Se utiliza para tomar los primeros n elementos de la lista.

Definición del método: *deftake(n: Int): List[A]*
Donde, n es el número de elementos a tomar de la lista.

Tipo de retorno: *Devuelve una lista que contiene solo los primeros n elementos de la lista indicada o devuelve la lista completa si n es mayor que el número de elementos en la lista dada, utilizamos foreach para leer las filas de la lista take*



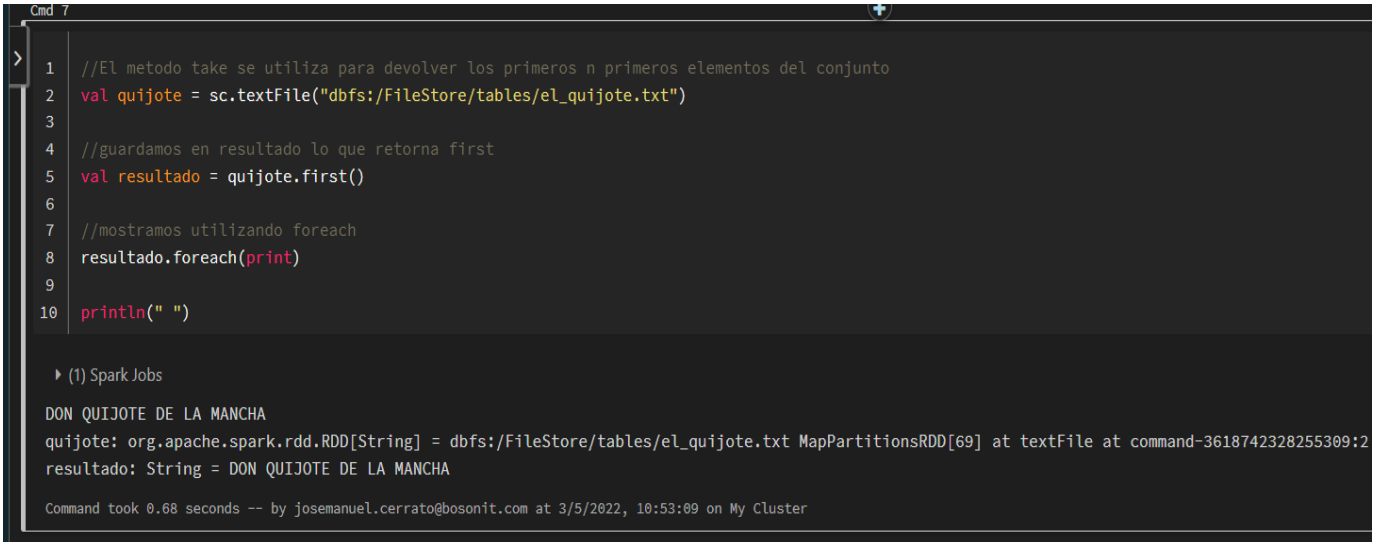
```
1  //El metodo take se utiliza para devolver los primeros n primeros elementos del conjunto
2  val quijote = sc.textFile("dbfs:/FileStore/tables/el_quijote.txt")
3
4  //guardamos en resultado lo que retorna take
5  val resultado = quijote.take(4)
6
7  //mostramos utilizando foreach
8  resultado.foreach(println)
9
10 println(" ")
```

```
▸ (1) Spark Jobs
DON QUIJOTE DE LA MANCHA
Miguel de Cervantes Saavedra

PRIMERA PARTE

quijote: org.apache.spark.rdd.RDD[String] = dbfs:/FileStore/tables/el_quijote.txt MapPartitionsRDD[67] at textFile at command-3618742328255308:2
resultado: Array[String] = Array(DON QUIJOTE DE LA MANCHA, Miguel de Cervantes Saavedra, "", PRIMERA PARTE)
Command took 0.62 seconds -- by josemanuel.cerrato@bosonit.com at 3/5/2022, 10:53:09 on My Cluster
```

Nos devuelve el primer elemento del conjunto de datos. Similar a take(X)
al igual que take hay que mostrarlo con un bucle ya que devuelve una lista



```
Cmd 7

1  //El metodo take se utiliza para devolver los primeros n primeros elementos del conjunto
2  val quijote = sc.textFile("dbfs:/FileStore/tables/el_quijote.txt")
3
4  //guardamos en resultado lo que retorna first
5  val resultado = quijote.first()
6
7  //mostramos utilizando foreach
8  resultado.foreach(print)
9
10 println(" ")

▸ (1) Spark Jobs
DON QUIJOTE DE LA MANCHA
quijote: org.apache.spark.rdd.RDD[String] = dbfs:/FileStore/tables/el_quijote.txt MapPartitionsRDD[69] at textFile at command-3618742328255309:2
resultado: String = DON QUIJOTE DE LA MANCHA
Command took 0.68 seconds -- by josemanuel.cerrato@bosonit.com at 3/5/2022, 10:53:09 on My Cluster
```