

# V Big Data and Machine Learning Bootcamp

## Data Processing

### PRACTICA

1. Hemos de crear una estructura de Kafka-Streaming PRODUCTOR - CONSUMIDOR de dos formas.

1.1 Usando los métodos de Kafka propios para ello.

En este caso, comprobaremos que si enviamos un fichero .json dado (personal.json) por el productor, el contenido de este fichero nos aparece en el consumidor.

1.2 Creando el código en Scala para montar un productor y un consumidor.

En este caso, ademas de comprobar el envío - recepción de este fichero, deberá contar el número de palabras del fichero y deberá filtrar (que no aparezcan) dos palabras a elegir. El resultado del .json filtrado se mostrará por consola.

2. Instalar Zepelin y configurarlo con Spark.

2.1 Mostrar los pasos necesarios para la instalación.

2.2 Calcular el número de registros que tenemos en el fichero amigos.csv

## SOLUCION

Para la resolución de la práctica, he instalado en local en mi ordenador MAC todas las herramientas necesarias, por lo que los comandos no coinciden exactamente con los vistos en clase, pero son los necesarios para la realización de la práctica.

Las herramientas instaladas son:

- Zookeeper 3.4.14
- Kafka 2.4.0
- Spark 2.4.4
- Scala
- IntelliJ IDEA 2019.3.2

### Punto 1.1

- Nos desplazamos al directorio donde se encuentran los ejecutables de Kafka

```
$ cd /usr/local/Cellar/kafka/2.4.0/bin
```

- Lanzamos el servidor de zookeeper

```
$ sudo zookeeper-server-start /usr/local/etc/kafka/zookeeper.properties
```

- A continuación, lanzamos el servidor de kafka

```
$ sudo kafka-server-start /usr/local/etc/kafka/server.properties
```

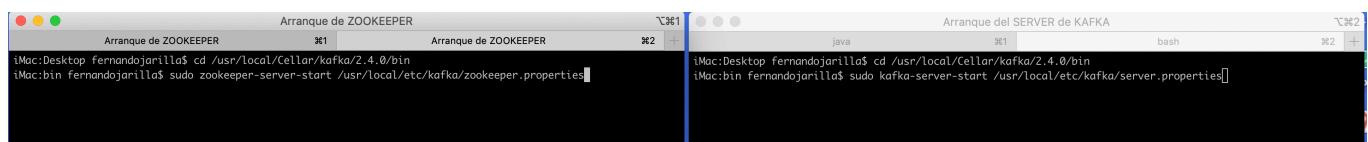


Figura 1. Comandos antes de ejecutarse\*

The image shows two terminal windows side-by-side. The left window is titled 'Arranque de ZOOKEEPER' and the right is 'Arranque del SERVER de KAFKA'. Both windows display command-line logs from their respective servers.

```

[2020-02-04 12:02:06.835] INFO Server environment:java.library.path=/Users/Fernandojarilla/Library/Java/Extensions:/Library/Java/Extensions:/System/Library/Java/Extensions:/usr/lib/java: (org.apache.zookeeper.server.ZooKeeperServer)
[2020-02-04 12:02:06.836] INFO Server environment:java.io.tmpdir=/var/folders/zz/zyxvpxq6cfsfvn_m0000000000000000/T/ (org.apache.zookeeper.server.ZooKeeperServer)
[2020-02-04 12:02:06.836] INFO Server environment:java.compiler=<NA> (org.apache.zookeeper.server.ZooKeeperServer)
[2020-02-04 12:02:06.836] INFO Server environment:os.name=Mac OS X (org.apache.zookeeper.server.ZooKeeperServer)
[2020-02-04 12:02:06.836] INFO Server environment:os.arch=x86_64 (org.apache.zookeeper.server.ZooKeeperServer)
[2020-02-04 12:02:06.836] INFO Server environment:os.version=10.14.6 (org.apache.zookeeper.server.ZooKeeperServer)
[2020-02-04 12:02:06.836] INFO Server environment:user.name=root (org.apache.zookeeper.server.ZooKeeperServer)
[2020-02-04 12:02:06.836] INFO Server environment:user.home=/var/root (org.apache.zookeeper.server.ZooKeeperServer)
[2020-02-04 12:02:06.836] INFO Server environment:user.dir=/usr/local/Cellar/kafka/2.4.0/bin (org.apache.zookeeper.server.ZooKeeperServer)
[2020-02-04 12:02:06.836] INFO Server environment:os.memory.free=497MB (org.apache.zookeeper.server.ZooKeeperServer)
[2020-02-04 12:02:06.836] INFO Server environment:os.memory.max=512MB (org.apache.zookeeper.server.ZooKeeperServer)
[2020-02-04 12:02:06.836] INFO Server environment:os.memory.total=512MB (org.apache.zookeeper.server.ZooKeeperServer)
[2020-02-04 12:02:06.840] INFO minSessionTimeout set to 6000 (org.apache.zookeeper.server.ZooKeeperServer)
[2020-02-04 12:02:06.840] INFO maxSessionTimeout set to 60000 (org.apache.zookeeper.server.ZooKeeperServer)
[2020-02-04 12:02:06.840] INFO Created server with tickTime 30000 minSessionTimeout 60000 maxSessionTimeout 600000 datadir /usr/local/var/lib/zookeeper/version-2 snapdir /usr/local/var/lib/zookeeper/version-2 (org.apache.zookeeper.server.ZooKeeperServer)
[2020-02-04 12:02:06.934] INFO Using org.apache.zookeeper.server.NIOServerCnxnFactory as server connection factory (org.apache.zookeeper.server.ServerCnxnFactory)
[2020-02-04 12:02:06.959] INFO Configuring NIO connection handler with 10s sendable connection timeout, 2 selector thread(s), 16 worker threads, and 64 KB direct buffers. (org.apache.zookeeper.server.NIOServerCnxnFactory)
[2020-02-04 12:02:07.022] INFO binding to port 0.0.0.0/0.0.0.0:2181 (org.apache.zookeeper.server.NIOServerCnxnFactory)
[2020-02-04 12:02:07.063] INFO zookeeper.snapshotSizeFactor = 0.33 (org.apache.zookeeper.server.ZKDatabase)
[2020-02-04 12:02:07.112] INFO Reading snapshot /usr/local/var/lib/zookeeper/version-2/snapshot.0 (org.apache.zookeeper.server.persistence.FileSnap)
[2020-02-04 12:02:07.216] INFO Snapshottting: 0xa0 to /usr/local/var/lib/zookeeper/version-2/snapshot.a0 (org.apache.zookeeper.server.persistence.FileTxnSnapshotLog)
[2020-02-04 12:02:07.307] INFO Using checkIntervalMs=60000 maxPerMinute=10000 (org.apache.zookeeper.server.Committer)
[2020-02-04 12:04:30,118] INFO Creating new log file: log.a3 (org.apache.zookeeper.server.persistence.FileTxnLog)

```

[2020-02-04 12:04:32.004] INFO [GroupMetadataManager brokerId=0] Finished loading offsets and group metadata from \_consumer\_offsets-14 in 0 milliseconds. (kafka.coordinator.group.GroupMetadataManager)

[2020-02-04 12:04:32.004] INFO [GroupMetadataManager brokerId=0] Finished loading offsets and group metadata from \_consumer\_offsets-20 in 0 milliseconds. (kafka.coordinator.group.GroupMetadataManager)

[2020-02-04 12:04:32.004] INFO [GroupMetadataManager brokerId=0] Finished loading offsets and group metadata from \_consumer\_offsets-23 in 0 milliseconds. (kafka.coordinator.group.GroupMetadataManager)

[2020-02-04 12:04:32.005] INFO [GroupMetadataManager brokerId=0] Finished loading offsets and group metadata from \_consumer\_offsets-26 in 0 milliseconds. (kafka.coordinator.group.GroupMetadataManager)

[2020-02-04 12:04:32.005] INFO [GroupMetadataManager brokerId=0] Finished loading offsets and group metadata from \_consumer\_offsets-17 in 0 milliseconds. (kafka.coordinator.group.GroupMetadataManager)

[2020-02-04 12:04:32.006] INFO [GroupMetadataManager brokerId=0] Finished loading offsets and group metadata from \_consumer\_offsets-29 in 0 milliseconds. (kafka.coordinator.group.GroupMetadataManager)

[2020-02-04 12:04:32.006] INFO [GroupMetadataManager brokerId=0] Finished loading offsets and group metadata from \_consumer\_offsets-35 in 0 milliseconds. (kafka.coordinator.group.GroupMetadataManager)

[2020-02-04 12:04:32.006] INFO [GroupMetadataManager brokerId=0] Finished loading offsets and group metadata from \_consumer\_offsets-38 in 0 milliseconds. (kafka.coordinator.group.GroupMetadataManager)

[2020-02-04 12:04:32.006] INFO [GroupMetadataManager brokerId=0] Finished loading offsets and group metadata from \_consumer\_offsets-12 in 0 milliseconds. (kafka.coordinator.group.GroupMetadataManager)

[2020-02-04 12:04:32.007] INFO [GroupMetadataManager brokerId=0] Finished loading offsets and group metadata from \_consumer\_offsets-15 in 1 milliseconds. (kafka.coordinator.group.GroupMetadataManager)

[2020-02-04 12:04:32.007] INFO [GroupMetadataManager brokerId=0] Finished loading offsets and group metadata from \_consumer\_offsets-18 in 1 milliseconds. (kafka.coordinator.group.GroupMetadataManager)

[2020-02-04 12:04:32.007] INFO [GroupMetadataManager brokerId=0] Finished loading offsets and group metadata from \_consumer\_offsets-21 in 0 milliseconds. (kafka.coordinator.group.GroupMetadataManager)

[2020-02-04 12:04:32.007] INFO [GroupMetadataManager brokerId=0] Finished loading offsets and group metadata from \_consumer\_offsets-24 in 0 milliseconds. (kafka.coordinator.group.GroupMetadataManager)

[2020-02-04 12:04:32.008] INFO [GroupMetadataManager brokerId=0] Finished loading offsets and group metadata from \_consumer\_offsets-27 in 0 milliseconds. (kafka.coordinator.group.GroupMetadataManager)

[2020-02-04 12:04:32.008] INFO [GroupMetadataManager brokerId=0] Finished loading offsets and group metadata from \_consumer\_offsets-30 in 0 milliseconds. (kafka.coordinator.group.GroupMetadataManager)

[2020-02-04 12:04:32.008] INFO [GroupMetadataManager brokerId=0] Finished loading offsets and group metadata from \_consumer\_offsets-33 in 1 milliseconds. (kafka.coordinator.group.GroupMetadataManager)

[2020-02-04 12:04:32.009] INFO [GroupMetadataManager brokerId=0] Finished loading offsets and group metadata from \_consumer\_offsets-36 in 0 milliseconds. (kafka.coordinator.group.GroupMetadataManager)

[2020-02-04 12:04:32.009] INFO [GroupMetadataManager brokerId=0] Finished loading offsets and group metadata from \_consumer\_offsets-39 in 0 milliseconds. (kafka.coordinator.group.GroupMetadataManager)

[2020-02-04 12:04:32.030] INFO [GroupCoordinator 0]: Loading group metadata for console-consumer-14024 with generation 2 (kafka.coordinator.group.GroupCoordinator)

[2020-02-04 12:04:32.031] INFO [GroupMetadataManager brokerId=0] Finished loading offsets and group metadata from \_consumer\_offsets-42 in 22 milliseconds. (kafka.coordinator.group.GroupMetadataManager)

[2020-02-04 12:04:32.031] INFO [GroupMetadataManager brokerId=0] Finished loading offsets and group metadata from \_consumer\_offsets-45 in 0 milliseconds. (kafka.coordinator.group.GroupMetadataManager)

[2020-02-04 12:04:32.031] INFO [GroupMetadataManager brokerId=0] Finished loading offsets and group metadata from \_consumer\_offsets-48 in 0 milliseconds. (kafka.coordinator.group.GroupMetadataManager)

Figura 2. Comandos de arranque se servers ejecutados

- Una vez lanzados los servers, crearé un tópico que llamaré **práctica**

```
$ sudo kafka-topics --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic practica
```

The terminal window shows the command being run:

```
Last login: Tue Feb 4 17:04:41 on ttys002
iMac:bin fernandojarilla$ sudo kafka-topics --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic practica
Password:
Created topic practica.
iMac:bin fernandojarilla$
```

Figura 3. Crear tópico

- A continuación, crearé el **PRODUCTOR**

```
$ sudo kafka-console-producer --broker-list localhost:9092 --topic practica
```

- Y finalmente el **CONSUMIDOR**

```
$ sudo kafka-console-consumer --bootstrap-server localhost:9092 --topic practica --from-beginning
```

The image shows two terminal windows side-by-side. The left window is titled 'CREAR PRODUCTOR' and the right is 'CREAR CONSUMIDOR'. Both windows display command-line logs from their respective components.

[Mac:bin fernandojarilla\$ sudo kafka-console-producer --broker-list localhost:9092 --topic practica]

iMac:bin fernandojarilla\$ sudo kafka-console-consumer --bootstrap-server localhost:9092 --topic practica --from-beginning

Figura 4. Lanzados productor y consumidor

- Por fin, lo que escriba en la ventana del productor, aparecerá en la ventana del consumidor. La información se envía tras pulsar *Return* en el productor

The screenshot shows two terminal windows side-by-side. The left window, titled 'CREAR PRODUCTOR', contains three tabs labeled 'java', 'java', and 'java'. The right window, titled 'CREAR CONSUMIDOR', also has three tabs labeled 'java', 'java', and 'java'. Both windows show the command 'sudo kafka-console-producer --broker-list localhost:9092 --topic practica' followed by the message 'Hola buenos días' and 'Enviando información'. The 'Consumidor' window shows the message 'Hola buenos días' and 'Enviando información'.

Figura 5. Primeros datos enviados y segunda linea antes de enviarse

This screenshot is similar to Figure 5, but the 'Consumidor' window now shows the message 'Enviando información' on a new line, indicating the second line of data has been received.

Figura 6. Segunda línea enviada

This screenshot shows additional data being sent. The 'Consumidor' window displays the message 'Con esto se comprueba el punto 1 de la práctica' followed by 'lanzo algo por el productor y me aparece en el consumidor.'

Figura 7. Envío de algo mas de información

- Finalmente, para enviar el fichero .json salgo del productor (con *Ctrl+C*) y lanzo el siguiente comando.

```
$ cat /users/fernandojarilla/Desktop/personal.json | sudo kafka-console-producer --broker-list localhost:9092 --topic practica
```

The screenshot shows the 'Productor' window executing the command 'cat /users/fernandojarilla/Desktop/personal.json | sudo kafka-console-producer --broker-list localhost:9092 --topic practica'. The 'Consumidor' window shows the contents of the 'personal.json' file being printed line by line, starting with 'Hola buenos días' and 'Enviando información'.

```
{
  "id": 1,
  "first_name": "Jeanette",
  "last_name": "Pendreth",
  "email": "jpPENDRETH@census.gov",
  "gender": "Female",
  "ip_address": "26.58.193.2"
},
{
  "id": 2,
  "first_name": "Giovani",
  "last_name": "Frediani",
  "email": "gfrediani@note.net",
  "gender": "Male",
  "ip_address": "229.179.4.212"
},
{
  "id": 3,
  "first_name": "Noell",
  "last_name": "Boo",
  "email": "nbooz@imageershock.us",
  "gender": "Female",
  "ip_address": "180.66.162.255"
},
{
  "id": 4,
  "first_name": "Willard",
  "last_name": "Volek",
  "email": "wvolek@3rvk.com",
  "gender": "Male",
  "ip_address": "67.76.188.26"
}
```

Figura 8. Envio y visualización de la recepción del fichero personal.json

Con esto queda terminada la primera parte de la práctica.