

Datos del estudiante

Nombre y apellidos

Fecha de entrega

Actividad: Spark Streaming y Kafka

Objetivos de la actividad

Con esta actividad, los estudiantes pondrán en práctica los conocimientos estudiados en las clases de teoría acerca del manejo de Apache Spark (módulo *Streaming*) y Kafka. Completarán un ejercicio sencillo que involucra las dos tecnologías y que los ayudará a entender mejor el propósito de cada una y a verlas funcionando en un caso concreto.

Pautas de elaboración

A continuación, describimos el trabajo que debe llevar a cabo el alumno y presentaremos la infraestructura disponible en la que se realizarán las tareas, así como unas orientaciones sobre cómo utilizarla. Se compone de dos partes diferentes. En cada una, hemos desglosado la explicación en pasos para facilitar su resolución.

PARTE 1. Manejo de Spark Streaming. Las instrucciones se encuentran en el *notebook* actividad2.ipynb. Recuerda **subir este *notebook* al directorio GCS** que aparece en el menú lateral de JupyterLab. Después, una vez abierto el *notebook* por primera vez, has de cambiar la opción Python3 que aparece en la esquina superior derecha por PySpark, tal y como se describió en la Actividad 1.

PARTE 2. Manejo de Apache Kafka. En las instrucciones que se encuentran en el *notebook* anterior, se plantea un ejercicio para leer de un *topic* de Kafka utilizando Apache Spark. Una vez ejecutadas todas las celdas del *notebook*:

- Abrimos una terminal de Linux por SSH a la máquina <nombrecluster>-m:

The screenshot shows the Google Cloud Dataproc console interface. On the left, the 'Clusters' tab is selected in the sidebar. The main panel displays details for a cluster named 'unircluster'. The cluster is in a 'Running' status. Below the details, the 'VM INSTANCES' tab is selected, showing a table of instances. The table has columns for 'Name' and 'Role'. The instances listed are 'unircluster-m' (Master), 'unircluster-w-0' (Worker), and 'unircluster-w-1' (Worker). An 'SSH' button is visible next to the Master instance.

| Name | Role |
|-----------------|--------|
| unircluster-m | Master |
| unircluster-w-0 | Worker |
| unircluster-w-1 | Worker |

Figura 1. Terminal de Linux por SSH a la máquina. Fuente: elaboración propia.

- Creamos un nuevo *topic*, llamado «retrasos», ejecutando el siguiente comando en una sola línea:

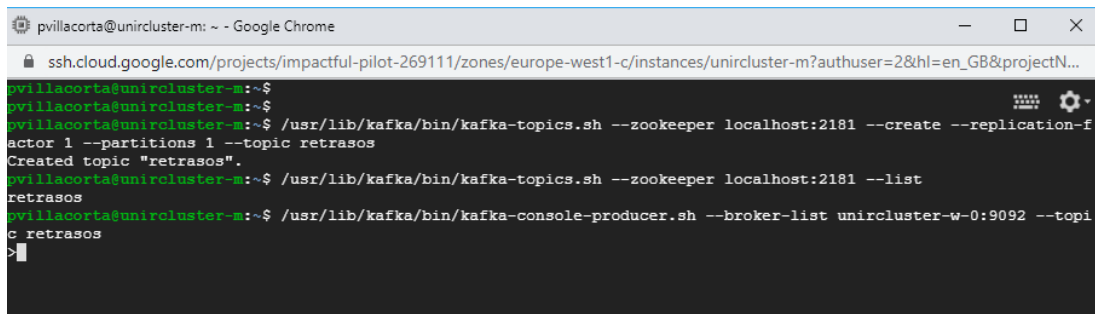
```
/usr/lib/kafka/bin/kafka-topics.sh --zookeeper localhost:2181 --create --replication-factor 1 --partitions 1 --topic retrasos
```

- Después podemos obtener la lista de *topics* existentes con el comando:

```
/usr/lib/kafka/bin/kafka-topics.sh --zookeeper localhost:2181 --list
```

- Ejecutamos el kafka-console-producer, que es un productor de Kafka que envía los mensajes que escribamos por teclado al *topic* que le indiquemos. Debemos cambiar el nombre del clúster (<nombrecluster>) por el que tengamos en cada caso (lo podéis consultar en la propia terminal, justo detrás de @: es el nombre que viene antes de «-m»):

```
/usr/lib/kafka/bin/kafka-console-producer.sh --broker-list <nombrecluster>-w-0:9092 --topic retrasos
```



```
pvillacorta@unircluster-m: ~ - Google Chrome
ssh.cloud.google.com/projects/impactful-pilot-269111/zones/europe-west1-c/instances/unircluster-m?authuser=28&hl=en_GB&projectN...
pvillacorta@unircluster-m:~$
pvillacorta@unircluster-m:~$
pvillacorta@unircluster-m:~$ /usr/lib/kafka/bin/kafka-topics.sh --zookeeper localhost:2181 --create --replication-factor 1 --partitions 1 --topic retrasos
Created topic "retrasos".
pvillacorta@unircluster-m:~$ /usr/lib/kafka/bin/kafka-topics.sh --zookeeper localhost:2181 --list
retrasos
pvillacorta@unircluster-m:~$ /usr/lib/kafka/bin/kafka-console-producer.sh --broker-list unircluster-w-0:9092 --topic retrasos
>
```

Figura 2. Ejecución de Kafka console. Fuente: elaboración propia.

El productor de consola suele utilizarse para desarrollo y testeo, pero nunca para entornos productivos. Tras cada mensaje, debemos pulsar ENTER. El nombre del bróker de Kafka al que va dirigido el mensaje puede ser tanto el *worker* 0 como el 1 de nuestro clúster. Lo que se indica es el nombre de la máquina, como `<nombre_cluster>-w-0` (o bien `w-1`).

- ▶ Cada mensaje debe ser una línea de texto con estructura JSON. La estructura y contenido de los mensajes están indicados en el *notebook*.
- ▶ Una vez tecleado un mensaje con la estructura anterior, hemos de pulsar ENTER para que el productor lo envíe a Kafka.
- ▶ El programa no admite borrado de caracteres, pero puede finalizarse en cualquier momento pulsando Ctrl + C, y volverse a ejecutar escribiendo de nuevo el comando anterior en la terminal de Linux que se abrió por SSH.
- ▶ Recomendamos que la captura de pantalla incluya tanto la ventana con los mensajes escritos en el `kafka-console-producer.sh` como el último *batch* actualizado por Spark Structured Streaming.

Entregables

- ▶ *Notebook* que funcione, donde todos los <COMPLETAR> hayan sido rellenados.
- ▶ Breve documento con las capturas de pantalla de haber ejecutado con éxito las escrituras realizadas en el productor de consola; sobre todo, captura de pantalla

de la salida mostrada en la consola por las sucesivas operaciones `show()` del DF resultante del procesado con Spark Structured Streaming.