KAFKA IOT

Jose Pablo Fernandez
Juan Francisco Galan

INDICE

KAFKA

Abrimos PoweShell, en nuestro caso nos conectamos a la direccion 172.17.10.33, clave usuario.
Paso 1: arrancamos zookeeper.properties
Paso 2: arrancamos server-31.properties (fichero principal de configuracion de kafka)

```
PS C:\Users\Pablo> ssh usuario@172.17.10.33
usuario@172.17.10.33's password:
```

```
usuario@ceserver3:~/kafka_2.13-3.1.0/bin$ zookeeper-server-start.sh -daemon ../config/zookeeper.properties
```

```
usuario@ceserver3:~/kafka_2.13-3.1.0/bin$ kafka-server-start.sh -daemon ../config/server-31.properties
```

Comprobamos que zookeeper esta arrancado con el comando ps aux | grep kafka

```
usuario@ceserver3:~/kafka_2.13-3.1.0/bin$ ps aux | grep kafka
usuario    94215  1.0  1.2 2631420 102456 pts/0  Sl   18:22   0:01 java -Xmx512M -Xms512M -server -XX:+UseG1GC -XX:Ma
xGCPauseMillis=20 -XX:InitiatingHeapOccupancyPercent=35 -XX:+ExplicitGCInvokesConcurrent -XX:MaxInlineLevel=15 -Djava
.awt.headless=true -Xlog:gc*:file=/home/usuario/kafka_2.13-3.1.0/bin/../logs/zookeeper-gc.log:time,tags:filecount=10,
filesize=100M -Dcom.sun.management.jmxremote -Dcom.sun.management.jmxremote.authenticate=false -Dcom.sun.management.j
mxremote.ssl=false -Dkafka.logs.dir=/home/usuario/kafka_2.13-3.1.0/bin/../logs -Dlog4j.configuration=file:/home/usuar
io/kafka_2.13-3.1.0/bin/../config/log4j.properties -cp /home/usuario/kafka_2.13-3.1.0/bin/../libs/activation-1.1.1.ja
r:/home/usuario/kafka_2.13-3.1.0/bin/../libs/aopalliance-repackaged-2.6.1.jar:/home/usuario/kafka_2.13-3.1.0/bin/../l
ibs/argparse4j-0.7.0.jar:/home/usuario/kafka_2.13-3.1.0/bin/../libs/audience-annotations-0.5.0.jar:/home/usuario/kafk
a_2.13-3.1.0/bin/../libs/commons-cli-1.4.jar:/home/usuario/kafka_2.13-3.1.0/bin/../libs/commons-lang3-3.8.1.jar:/home
/usuario/kafka_2.13-3.1.0/bin/../libs/connect-api-3.1.0.jar:/home/usuario/kafka_2.13-3.1.0/bin/../libs/connect-basic-
auth-extension-3.1.0.jar:/home/usuario/kafka_2.13-3.1.0/bin/../libs/connect-file-3.1.0.jar:/home/usuario/kafka_2.13-3
```

Comprobamos que zookeeper esta arrancado con el comando ps aux | grep zookeeper

```
usuario@ceserver3:~/kafka_2.13-3.1.0/bin$ kafka-topics.sh --list --bootstrap-server 172.17.10.35:9092
__consumer_offsets
camaraAULAateca
camaraAULAcisco
humedadAULA142
humedadAULA145
humedadAULA149
humedadAULA159
humedadAULAAteca
humedadAULAateca
```

Aqui creamos el primer topic con 1 particion factor de replicacion 3, el nombre del topic y direccion y puertos correspondientes.
Tendremos que hacer lo mismo con la finalidad de crear 4 topics.(Humedad, Ruido, Luminusidad, Temperatura)

```
usuario@ceserver3: $ kafka-topics.sh --create --partitions 1 --replication-factor 3 --topic 2024_145_humedad --bootstrap-server 172.17.10.35:9092
WARNING: Due to limitations in metric names, topics with a period ('.') or underscore ('_') could collide. To avoid issues it is best to use either, but not both.
Created topic 2024_145_humedad.
usuario@ceserver3: $ kafka-topics.sh --create --partitions 1 --replication-factor 3 --topic 2024_145_ruido --bootstrap-server 172.17.10.35:9092
WARNING: Due to limitations in metric names, topics with a period ('.') or underscore ('_') could collide. To avoid issues it is best to use either, but not both.
Created topic 2024_145_ruido.
```

Comprobamos que tenemos creados los topics con el comando --list.

```
usuario@ceserver3: $ kafka-topics.sh --list --bootstrap-server ceserver3:9092
2024_142_HUMEDAD
2024_142_LUMINOSIDAD
2024_142_RUIDO
2024_142_TEMPERATURA
2024_145_humedad
2024_145_luminosidad
2024_145_ruido
2024_145_temperatura
2024_149_humedad
2024_149_luminosidad
2024_149_ruido
2024_149_temperatura
```

Fin del apartado de kafka.

HDFS

Nos conectamos al namenode desde powershel usuario: 192.17.10.30, clave: usuario

```
PS C:\Users\Pablo> ssh usuario@172.17.10.30
usuario@172.17.10.30's password:
```

Inicializamos hdfs

```
usuario@CEPrueba: $ start-dfs.sh
Starting namenodes on [Master]
Starting datanodes
Starting secondary namenodes [CEPrueba]
```

Creamos una carpeta con el nombre Pro_IOT

```
usuario@CEPrueba: $ hdfs dfs -mkdir /Pro_IOT
```

Comprobamos si esta creada.

```
usuario@CEPrueba: $ hdfs dfs -mkdir /Pro_IOT
usuario@CEPrueba: $ hdfs dfs -ls /
Found 7 items
drwxr-xr-x   - usuario supergroup          0 2024-02-27 18:43 /Pro_IOT
drwxrwxr-x   - usuario supergroup          0 2023-05-11 21:04 /Proyecto
drwxr-xr-x   - usuario supergroup          0 2023-04-13 20:27 /flume01
drwxr-xr-x   - usuario supergroup          0 2023-04-17 19:23 /flume02
drwxr-xr-x   - usuario supergroup          0 2023-04-17 19:26 /flume03
drwxr-xr-x   - usuario supergroup          0 2023-05-10 19:25 /prueba
drwxr-xr-x   - usuario supergroup          0 2023-05-04 21:05 /user
```

Creamos dentro de /Pro_IOT otra carpeta con 145 y entro (ruido, temperatura y humedad)

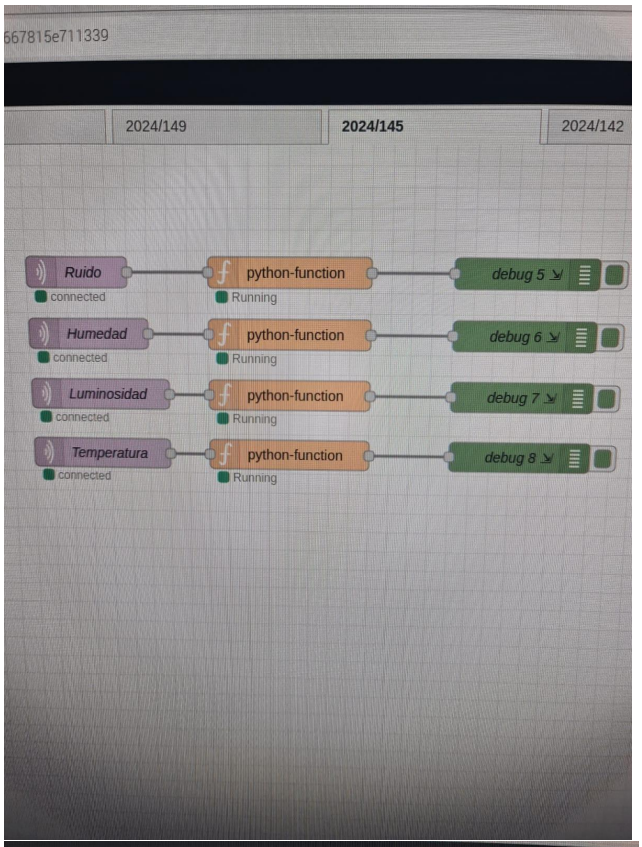
```
usuario@CEPrueba: $ hdfs dfs -mkdir /Pro_IOT/145
usuario@CEPrueba: $ hdfs dfs -mkdir /Pro_IOT/145/ruido
usuario@CEPrueba: $ hdfs dfs -mkdir /Pro_IOT/145/temperatura
usuario@CEPrueba: $ hdfs dfs -mkdir /Pro_IOT/145/luminosidad
usuario@CEPrueba: $ hdfs dfs -mkdir /Pro_IOT/145/humedad
```

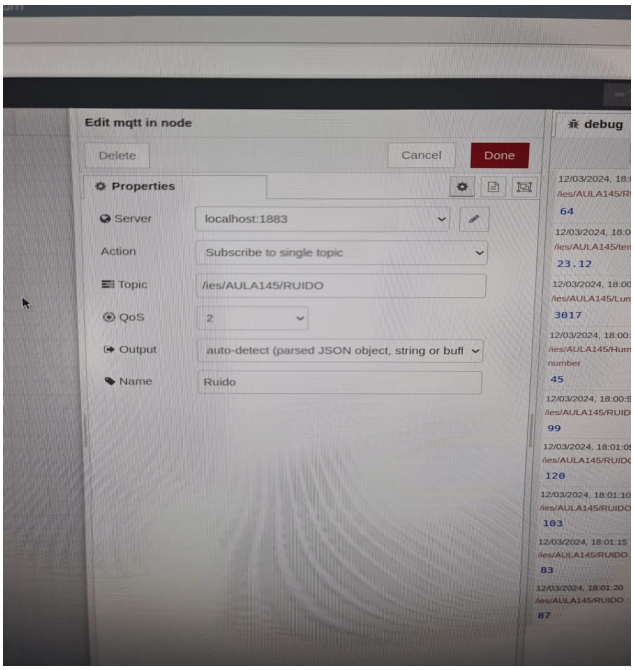
Comprobamos que se crea correctamente.

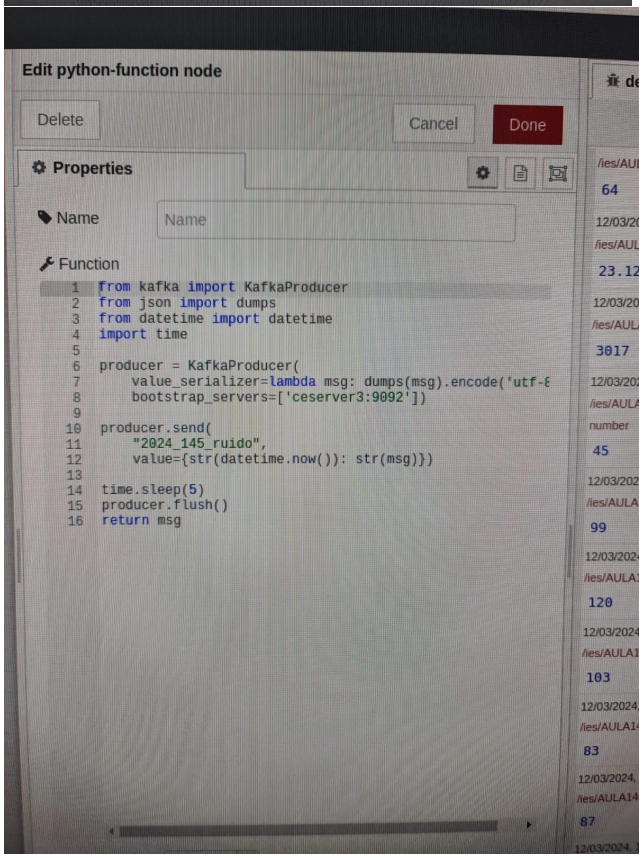
```
usuario@CEPrueba: $ hdfs dfs -ls /Pro_IOT/145
Found 4 items
drwxr-xr-x   - usuario supergroup          0 2024-02-27 19:10 /Pro_IOT/145/humedad
drwxr-xr-x   - usuario supergroup          0 2024-02-27 19:10 /Pro_IOT/145/luminosidad
drwxr-xr-x   - usuario supergroup          0 2024-02-27 19:09 /Pro_IOT/145/ruido
drwxr-xr-x   - usuario supergroup          0 2024-02-27 19:09 /Pro_IOT/145/temperatura
usuario@CEPrueba: $
```

En la parte de IOT hemos creado los productores.

PRODUCTORES







CONSUMIDORES

```
# Importación de las bibliotecas necesarias
from hdfs import InsecureClient  # Para interactuar con HDFS
from kafka import KafkaConsumer  # Para consumir mensajes de Kafka
import time  # Para manejar operaciones de tiempo
import re  # Para realizar operaciones de búsqueda de patrones en texto
import os  # Para interactuar con el sistema operativo

# Datos de conexión para HDFS
HDFS_HOSTNAME = '172.17.10.30'  # Dirección IP del servidor HDFS
HDFSCLI_PORT = 9870  # Puerto de conexión para HDFS
HDFSCLI_CONNECTION_STRING = f'http://{HDFS_HOSTNAME}:{HDFSCLI_PORT}'  # Cadena de
conexión para HDFS

# Inicialización de variables
contador = 0  # Contador para controlar el número de registros procesados
ficheroCreado = False  # Bandera para controlar si se ha creado el archivo local

# Creación de un cliente HDFS no seguro
hdfs_client = InsecureClient(HDFSCLI_CONNECTION_STRING)

# Configuración del consumidor de Kafka
consumer = KafkaConsumer(
   '2024_145_ruido',  # Tópico de Kafka desde el cual se consumen los mensajes
   enable_auto_commit=True,  # Habilitar la confirmación automática de los mensajes consumidos
   bootstrap_servers=['172.17.10.35:9092', '172.17.10.34:9092', '172.17.10.33:9092']  # Servidores de
Kafka a los que se conectará el consumidor
)

# Bucle para recibir y procesar mensajes de Kafka
for m in consumer:
   # Extracción de datos del mensaje Kafka
   json_value = str(m.value)  # Convertir el mensaje a una cadena
   fecha = json_value.split('\')[0]  # Extraer la fecha del mensaje
   fecha = fecha.split('"')[1]  # Limpiar el formato de la fecha

   ruido_sucio = json_value.split("\")[6]  # Extraer el valor del ruido del mensaje
   ruido_valor = re.findall(r'\d+', ruido_sucio)  # Encontrar los números en el valor del ruido
   ruido = ''.join(ruido_valor)  # Unir los números encontrados en una cadena

   # Construcción del registro con la fecha y el valor del ruido
   registro = f'{fecha};{ruido}'
   print(registro)  # Imprimir el registro procesado

   # Verificar si se debe escribir el registro en un archivo local
   if contador <= 60:
      if not ficheroCreado:  # Verificar si el archivo aún no ha sido creado
         # Generar un nombre de archivo basado en la fecha actual y la hora
         fecha_actual = time.localtime()
         fecha_actual_con_hora = time.strftime("%Y-%m-%d %H-%M ruido", fecha_actual)
         nom_fichero = f"{fecha_actual_con_hora}.csv"

         # Crear el archivo y escribir el primer registro
         with open(nom_fichero, 'a') as archivo:
            archivo.write(f'{registro}\n')
```

```
            ficheroCreado = True  # Actualizar la bandera de archivo creado

        contador += 1  # Incrementar el contador de registros procesados

        # Escribir el registro en el archivo
        with open(nom_fichero, 'a') as archivo:
            archivo.write(f'{registro}\n')

    else:  # Si se han procesado más de 60 registros
        # Escribir el archivo en HDFS y eliminarlo del sistema local
        hdfs_client.write(f"/Pro_IOT/145/ruido/{nom_fichero}", nom_fichero)
        os.remove(nom_fichero)  # Eliminar el archivo local
```