**1.- Utilizando la sintaxis de Kafka-console-consumer.sh, comprobar que el consumidor puede leer el fichero json enviado por el productor.**

1.- Creamos el topic

Bin/Kafka-topics.sh –create –zookeeper localhost:2181 –replication-factor 1 –partitions 1 –topic nametopic



2.- Guardamos el fichero personal.json en el directorio bin de Kafka.



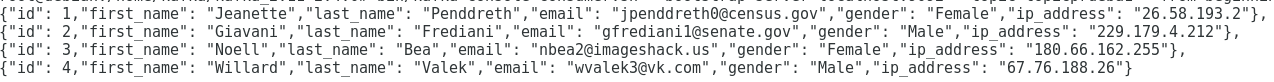
3.- Lanzamos el consumidor



4.- Lanzamos el productor



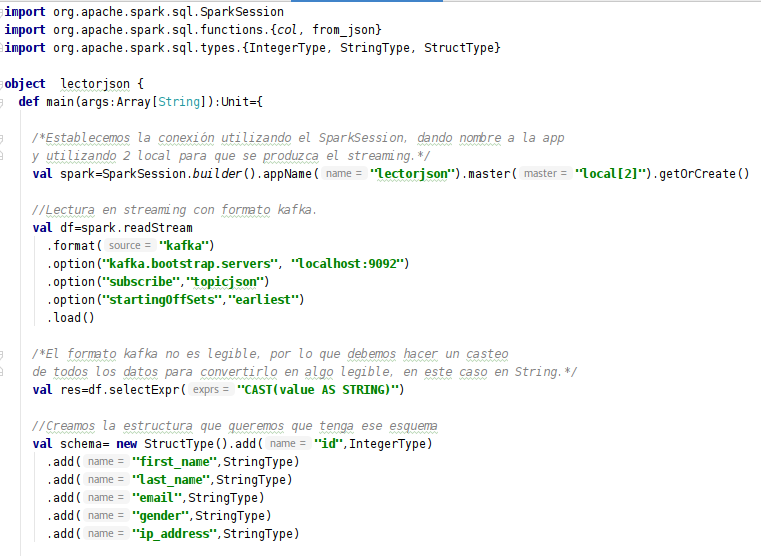
5 Visualización de datos en tiempo real en el consumidor.

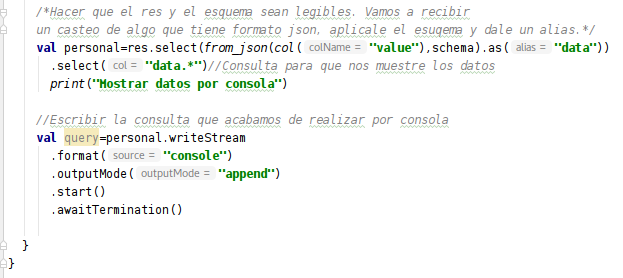


**2.- Creando un código en Scala, comprobar que el consumidor puede leer el fichero json enviado por el productor.**

En este apartado, se hace lo mismo que en el apartado 1 pero con código en Scala. Introducimos los datos desde el productor en la terminal y son leídos por un consumidor creado por un código en Scala, donde mediante una consulta se mostrarán los datos en la consola.

A continuación, adjunto los recortes de pantalla del código del consumidor en Scala con sus respectivos comentarios.





Una vez tenemos el código del consumidor listo, para que la aplicación empiece a funcionar debemos:

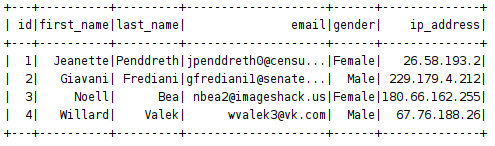
1.- Crear un topic. En este caso utilizamos el topicprueba3.

2.- Lanzamos el consumidor en Scala.

3.- Lanzamos el productor desde la terminal, con el fichero que queremos que se muestre en la consola.



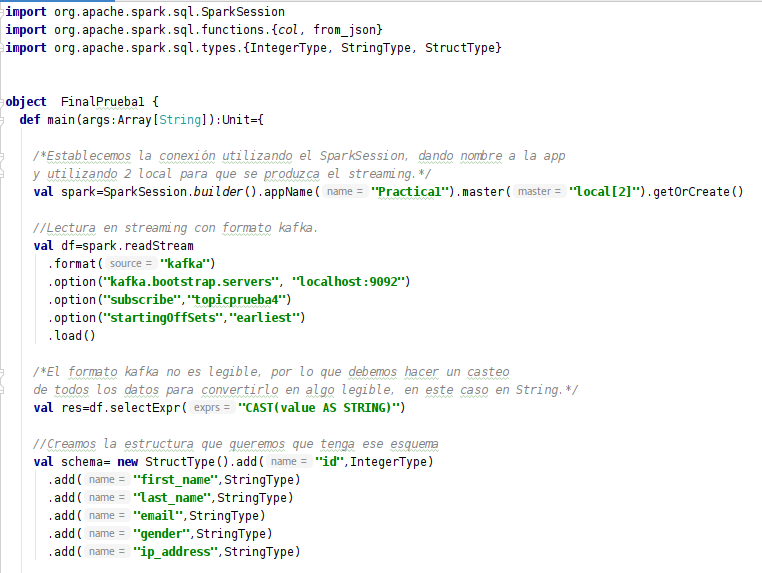
4. Resultados de la consulta que lanzamos para que nos muestren los datos que leímos del fichero mandado por el productor.

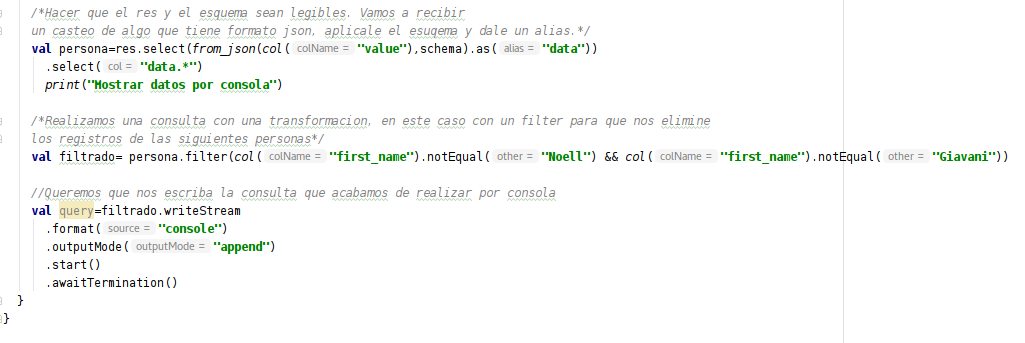


**3.- Ampliar el código del apartado 2 para que, a parte de leer el fichero realice un tratamiento especial de los datos que vamos leyendo. En este caso, queremos que filtre del fichero JSON dos palabras que elegimos cada uno.**

En este caso, utilizamos el código del apartado 2 y realizamos una consulta con una transformación para borrar los nombres Noell y Giavani. Por lo que, introduciremos el fichero json al completo y solamente nos mostraran en pantalla los 2 registros que no tienen los nombres anteriores.

A continuación, adjunto los recortes de pantalla del código del consumidor en Scala con sus respectivos comentarios.





Una vez tenemos el código del consumidor listo, para que la aplicación empiece a funcionar debemos:

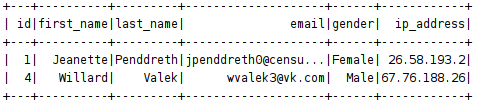
1.- Crear un topic. En este caso utilizamos el topicprueba4.

2.- Lanzamos el consumidor en Scala.

3.- Lanzamos el productor desde la terminal, con el fichero que queremos que se muestre en la consola.



4. Resultados de la consulta que lanzamos, donde podemos ver que se han eliminado los dos nombre establecidos.



**4.- Parte de investigación. Instalación de zeppelin en la máquina y realizar una consulta para calcular el número de registros introducidos.**

**1.- Instalación de zeppelin.**

Lo primero que debemos hacer es descomprimir el fichero zeppelin-0.8.2-bin-all.tgz:



Añadimos una nueva entrada en el fichero .bashrc, donde definimos donde se encuentra:



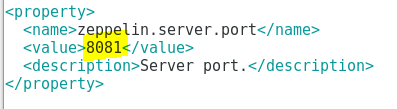


Cargamos las variables definidas anteriormente en el fichero .bashrc:



Por último, cambiamos el puerto por defecto (8080), ya que también es usado por Spark.





Iniciamos Zeppelin:

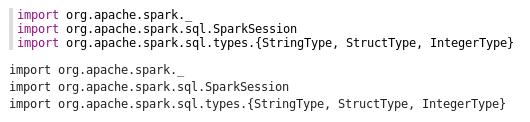


Una vez ya estamos en zeppelin en localhost:8081, creamos un nuevo interpreter llamado spark2 para poder lanzar consultas con la versión de spark 2.4.4

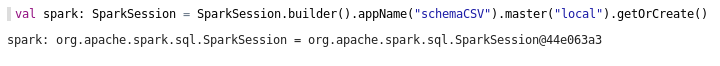


**2.- Calcular el número de registros en el fichero amigos.csv**

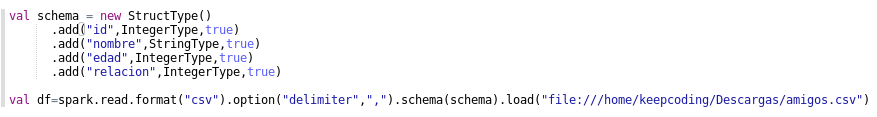
1.- Importamos las librerías necesarias.



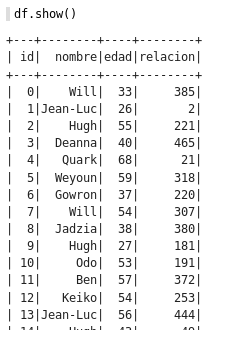
2.- Establecemos la conexión con SparkSession.



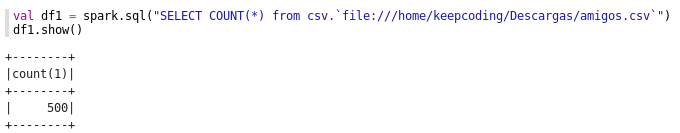
3.- El fichero no tiene cabecera, por lo que tenemos que crear un schema para poder darle una. Una vez establecida creamos el dataframe, leemos el fichero con formato csv, delimitado por comas y le decimos que use el esquema creado. Cargamos el fichero, para ello debemos decirle la ubicación del mismo:



4.- Mostramos los datos guardados en el dataframe. Como vemos es el fichero amigos.csv con las columnas creadas con el schema.



5.- Mediante una consulta lanzada con sparq sql, realizamos un count para ver cuántos registros tenemos.



Adjunto enlace de zeppelin con el código:

<http://localhost:8081/#/notebook/2F224SU2G>