Práctica Big Data Processing (IV)

Octubre 2019 / 1.0

Contexto:

Siglo XXIII, los ciudadanos y sus comunicaciones son espiados por el Gobierno de un país un tanto particular.

Introducción:

En secreto, instituciones ocultas del gobierno de *Cloacalandia* espían desde hace tiempo a los ciudadanos de las grandes ciudades del país. Estos usuarios utilizan una red social llamada *Celebram* enviando mensajes a sus conocidos y familiares. Los mensajes son cifrados por la red social, pero esto no es problema para los hackers del departamento del ministerio ya que han diseñado un algoritmo que es capaz de descifrar todos los mensajes. Una vez se envían, son 'esnifados' por dispositivos IOT ocultos y repartidos por diversas zonas de la ciudad generando **información de forma ininterrumpida** (lo que llamamos 'streaming'). Todos los mensajes, al ser interceptados, son marcados por un huella temporal (timestamp), ademas de añadir la zona desde donde han sido 'ingestados' (zona del iot)

Hemos sido contratados para crear un algoritmo que ayude a este gobierno. Dejamos a un lado los escrúpulos y elegimos pensar en los honorarios. (quizás tengamos alma de mercenarios…)

De vosotros, como desarrolladores Big Data, se requiere…

1.- Crear el esquema de cada uno de los datasets (ver orientación)
2.- Rellenarlos con info dummy según esquema
2.- Con el fin de procesar en tiempo real toda la información, tendréis que conseguir elaborar una única fuente de información 'completa' con la que trabajar, por lo previamente habréis tenido que preparar la info, quitar duplicados (si los hubiere), agrupar, ordenar, etc y todo aquello que creáis necesario para el correcto y posterior proceso.
El fin último es hallar por hora (ventana temporal) **las 10 palabras más usadas en los mensajes de tal ventana**. Una vez realizado este proceso, en caso de que la palabra más repetida coincida con alguna de las palabras de la lista negra, el sistema (nuestra aplicación) deberá enviar una notificación al ministro avisando de tal situación.

Partimos de la base de que…

- El sistema funciona 24/7
- Algunos IOT pueden dejar de funcionar, o bien por batería o bien porque se apaguen en remoto (estado apagado). Los IOT apagados no deberán contabilizar para la ingesta de datos.
- Los sistemas de notificación serán simulados
- El sistema de 'desencriptado' (función) será simulado
- La lista negra existirá realmente, y deberéis hacer la comprobación de pertenencia a dicha lista por parte de la palabra más repetida.
- La palabra repetida no podrá ser una preposición ni conjunción ni artículo.
- Para simular el envío de datos por parte de los IOT's, enchufaremos ficheros de texto para que los procese Kafka.

Fuentes de datos y esquemas propuestos (orientativo):

MensajesCapturados:
Mensaje_Id (String)
Contenido (String)
User_Id (String)

Usuarios registrados:
User_Id (String)
Nombre (String)
Apellido (String)
Edad (Int)
Sexo (String)

Dispositivos IOT:
IoT_Id
Encendido(Bool)
Zona_Id (String)

ListaNegraPalabras(*):
Palabra (String)

(*) No es bigdata, volumen pequeño

Parte opcional

Si queréis cobrar un plus por vuestro trabajo, deberéis extraer la información de cómo se 'relacionan' en la red social algunos de los usuarios (GraphX).

Fuente de datos y esquema:

Conexiones:

Origen_Id (String)
Destino_Id (String)
Tipo_Conexion (String)

Se valorará…

- Claridad en el código
- Favorecer la legibilidad por encima de un código compacto
- El correcto uso de las APIs sql
- Explicación (comentada en el propio código) el porqué de cada una de las decisiones tomadas
- Uso preferido de spark sql, aunque no obligatorio
- Investigación y ensayo autodidacta de Scala
- ***Para sacar un buen resultado de la práctica no es estrictamente necesario que ésta funcione de forma completa mientras sí cumpla la mayoría de los puntos anteriores de valoración.***

Esta versión podrá ser modificada si el profesor o cualquier alumno encuentra un error o/y omisión. Cualquier cambio se comunicará oportunamente