# BigData

LABORATORIO BIDDATA

## Temario del laboratorio

- Introducción a  Apache Spark Stream
- Apache Spark Streaming

Recursos:

https://spark.apache.org/docs/latest/structured-streaming-programming-guide.html

## Introducción a Apache Spark Stream

Spark Streaming es la extensión de la API central de Apache Spark para aplicaciones de transmisión. Permite la ingesta de datos en tiempo real (secuencia no unida) que luego pueden ser procesados, almacenados o consumidos por otros procesos/aplicaciones. Spark Stream permite la escalabilidad, el alto rendimiento y el procesamiento de flujo tolerante a fallas de flujos de datos en vivo. Esos flujos de datos en vivo se pueden ser consumidos desde Kafka, Amazon Kinesis, Twitter e incluso desde sockets TCP.

El proceso por lo tanto se puede resumir en: 1) se recibe el flujo de datos de entrada 2) se divide en lotes o fragmentos. 3) los lotes de datos de entrada se envían al motor Spark donde se realiza la abstracción para transformarlos en Apache Spark Discretized Stream o, Spark DStream. Los DStreams se basan en Spark RDD, lo que les permite integrarse perfectamente con otros componentes de Apache Spark como Spark MLlib o Spark SQL.

## Arquitectura y ventajas de Spark Streaming

**Equilibrio de carga dinámico**:  Apache Spark permite que Dstreams realice una asignación detallada de recursos computacionales.
**Recuperación rápida de fallos y rezagados**:  los DStreams están habilitados para la asignación de recursos computacionales discretos que podrían ejecutarse en cualquier lugar.
**Unificación de análisis por lotes, streaming e interactivos**:  los DStreams se heredan de RDDS, lo que los hace interoperables a lo largo de las cargas de trabajo por lotes y streaming sin problemas.
**Análisis avanzados como aprendizaje automático y SQL interactivo:**  Dtreams aprovecha otras bibliotecas de Apache Spark como MLlib (aprendizaje automático), GraphX (análisis de gráficos), SQL,

Desarrollo

1) Análisis de registros de accesos de la NASA:

Registros de acceso de la NASA:

En este laboratorio, usaremos los registros de Nasa Access que se descargan de https://ita.ee.lbl.gov/html/contrib/NASA-HTTP.html o podemos descargar el mismo conjunto de datos de Kaggle (https://www.kaggle.com/datasets/souhagaa/nasa-access-log-dataset-1995) que se limpia para el proceso de transmisión.

Para este laboratorio, hemos utilizado el conjunto de datos de Kaggle. Tenemos una configuración de emulador de que transmite los registros de la NASA en el puerto 9999. Para podrá recibir estos registros cuando comience a trabajar en los archivos de Python debe lanzar el script de Python serverstreaming.py.

Este conjunto de datos tiene registros de solicitudes HTTPS en el servidor de la NASA en formato CSV y cada registro tiene atributos, que son los siguientes:

- row_index: número de secuencia de los registros en la base de datos
- host: haciendo la solicitud. Puede ser directamente el nombre de host o, en su defecto, la dirección de Internet.
- timestamp: la hora en que el host accedió al servidor de la NASA
- method: método de solicitud del host en forma HTTP (GET, POST o HEAD)
- url: devolver recurso ulr
- response: respuesta devuelta del servidor como códigos HTTP que pueden estar en el rango de 200 a 500 . Cada código HTTP tiene sus significados, como 200 que es " OK success " o 404 que indica "Not Found" .
- bytes in the reply. cantidad de los datos que habían sido devueltos

Modelo de programación

La spark streaming estructurada es una API de procesamiento de transmisión de alto nivel construida sobre el motor de procesamiento central de Apache Spark. Permite a los usuarios escribir consultas de transmisión de una manera similar a las consultas por lotes, lo que facilita la creación de aplicaciones de procesamiento de transmisión escalables y tolerantes a fallas. La transmisión estructurada permite a los usuarios expresar sus cálculos de transmisión como consultas estándar por lotes que operan en la tabla de entrada ilimitada. Su modelo de programación por lotes facilita escribir y razonar sobre consultas de

transmisión, mientras que su modelo de procesamiento incremental garantiza que los datos se procesen de manera correcta y confiable.

## Conceptos básicos

Considere el flujo de datos de entrada como la "Tabla de entrada". Cada elemento de datos que llega a la transmisión es como una nueva fila que se agrega a la Entrada

Una consulta sobre la entrada generará la "Tabla de resultados". Cada intervalo de activación (digamos, cada 1 segundo), se agregan nuevas filas a la Tabla de entrada, que eventualmente actualiza la Tabla de resultados. Siempre que se actualice la tabla de resultados, nos gustaría escribir las filas de resultados modificadas en un receptor externo.

Para este laboratorio, usaremos **sparkstreamingstarterkit.py**, que se encuentra en el enlace del laboratorio. Como ya tenemos los registros de la NASA transmitiendo en un puerto , solo tiene que conectarse a ese puerto y dirección específicos en los que se envían los registros, también hemos configurado la duración de la ventana en '50 segundos' y la duración deslizante = '30 segundos' para crear lotes de registros para realizar análisis. **Descargue el archivo sparkstreamingstarterkit. ejecutar el archivo, observar los resultados**. En el archivo dado, queries sobre dataframes que se crean a partir de tablas ilimitadas y por lotes para ver los resultados. Tenemos tres opciones para ello.

**Modo complete**: toda la tabla de resultados actualizada se escribirá en el almacenamiento externo. Depende del conector de almacenamiento decidir cómo manejar la escritura de toda la tabla.

**Modo de Append**: solo las filas nuevas añadidas en la tabla de resultados desde el último activador se escribirán en el almacenamiento externo. Esto es aplicable solo en las consultas en las que no se espera que cambien las filas existentes en la tabla de resultados.

**Modo de Update:** solo las filas que se actualizaron en la tabla de resultados desde el último activador se escribirán en el almacenamiento externo (disponible desde Spark 2.1.1). Tenga en cuenta que esto es diferente del modo completo en que este modo solo genera las filas que han cambiado desde el último activador. Si la consulta no contiene agregaciones, será equivalente al modo Agregar.

1) Debe ejecutar **sparkstreamingstarterkit** y ver los resultados. Obtendrá resultados, donde cada host tendrá dos entradas del lote actual y otra del lote anterior. Su tarea es usar otros métodos de visualización como se mencionó anteriormente, agregar y actualizar. Como referencia , use este enlace también https://spark.apache.org/docs/latest/structured-streaming-programming-guide.html

2) Cuente el código de acceso de respuesta HTTP que recibió cada host.

   De manera similar al proceso anterior, cuente el código de respuesta que cada host recibió del servidor. Por ejemplo, el código de respuesta http, 200, significa que la solicitud de acceso del host 001.msy4.communique.net se realizó 5 veces correctamente entre la ventana de tiempo. como referencia y ayuda, use este enlace https://sparkbyexamples.com/pyspark

3) Escribirá un código que sume el total de bytes de información que cada host descarga del servidor de la Nasa utilizando la propiedad de bytes.
   Nota: todas las propiedades están en cadena, cambie el tipo de datos según sea necesario.