UVG UNIVERSIDAD DEL VALLE DE GUATEMALA

CC-3080 Big Data

EJERCICIO No. 5
Spark Structured Streaming

En este ejercicio el estudiante se familiarizará con el concepto de Windows y podrá poner en práctica el uso de Structured Streaming.

Carga de Dataset

El instructor le proveerá el archivo spark_strmng_lab.zip. Descomprima y cargue el directorio completo (con subdirectorios) a una ubicación dentro de su Databricks.

Spark Structured Streaming

Entre los archivos recibidos, hay un directorio llamado **sensor_data** que contiene data de sensores de una bomba de agua en un lapso de 5 meses. La data está almacenada en formato json y contiene tomas por minuto de 52 diferentes sensores. Adicional, cuenta con un campo **machine_status** que indica el estado en el que se encuentra la bomba al momento de esta toma.

Vamos a simular un escenario en el que usted recibe la data de un flujo y debe calcular algunas estadísticas por hora para cada día que recibe y escribirlas en un directorio de output. Para lograrlo, el código que utilizará para cargar la data debe verse algo así (poner atención a la opción maxFilesPerTrigger):

```
> schema = spark.read.json(working_dir + '/sensor_data/').schema
> sensor_data = (spark
    .readStream
    .option("maxFilesPerTrigger", 1)
    .json(working_dir + '/sensor_data/', schema=schema)
  )
```

Aplique las transformaciones necesarias para generar un Streaming DataFrame que reporte lo siguiente:

- Valor máximo para sensor_00
- Valor mínimo para sensor_00
- Promedio del sensor_00
- Existió falla en el sistema

Aprovechando la implementación de Databricks, escriba sus resultados en una In-Memory-Table y consúltelos conforme se van generando. Tome el siguiente código como referencia para el Streaming Query:

```
> query = (report
    .writeStream
    .outputMode('complete')
    .format('memory')
    .queryName('reporte')
    .start()
  )
```

Widows

Dentro del directorio base encontrará un archivo **beach_weather.csv** que utilizaremos en esta primera parte del laboratorio.

Cargue la data en un DataFrame estático y utilizando **`functions.window()`** y otras funciones de la misma librería responda las siguientes preguntas:

- ¿En qué intervalo de 10 días corridos se reportó la máxima temperatura (Air Temperature) en cualquiera de los sensores?, ¿el máximo registrado en un momento de la data, se encuentra en ese intervalo? ¿Qué podemos decir de las lluvias más intensas (Rain Intensity)?
- Para las respuestas del punto anterior, ¿hay alguna tendencia año con año?
- En promedio, ¿cuál es el espacio de 3 horas del día en el que mayor radiación solar se percibe (Solar Radiation)?, ¿coincide con el de la temperatura?

Watermarks

Dentro del directorio base encontrará un subdirectorio con el nombre **lab03-data** y dentro de este encontrará unos archivos csv. También encontrará un directorio llamado **lab03-input** con únicamente un archivo, que corresponde al primero del directorio anterior.

Durante este laboratorio vamos a construir un proceso de Structured Streaming sobre el directorio lab03-input y manualmente vamos a ir depositando los archivos del lab03-data dentro del primero para que vayan siendo procesados. La data que se encuentra por archivo corresponde a un día completo, sin embargo estos han sido modificados intencionalmente para representar el atraso de información.

Lo primero que haremos será cargar los archivos de la carpeta lab03-data en un DataFrame tradicional (no streaming) y analizarlos. Cuando agrupamos la data por día (tomado de la columna Measurement Timestamp) y hacemos un count podemos ver la cantidad de registros por día.

```
>>> batch_df \
      .withColumn(
        'date',
        f.to_date('Measurement Timestamp', 'MM/dd/yyyy hh:mm:ss a')
      ).groupBy('date').count() \
      .show()
```

Podemos ver que hay 48 registros diarios, exceptuando los días 4 y 5 que les falta un poco de data.

```
>>> batch_df \
...   .withColumn(
...     'date',
...     f.to_date('Measurement Timestamp', 'MM/dd/yyyy hh:mm:ss a')
...   ) \
...   .groupBy('date').count() \
...   .show()
+----------+-----+
|      date|count|
+----------+-----+
|2020-06-04|   47|
|2020-06-05|   46|
|2020-06-03|   48|
|2020-06-02|   48|
|2020-06-01|   48|
+----------+-----+
```

Ahora crearemos un streaming DataFrame sobre la carpeta lab03-input, que en este momento sólo tiene un archivo. Sobre este DataFrame aplicaremos una agrupación en windows de 6 horas (tumbling) que vaya a hacer conteos sobre la data. En este momento antes de la agrupación es cuando debemos configurar un Watermark. Configuraremos uno de 24 horas:

```
>>> grouped = (
      data
      .withWatermark('timestamp', '24 hours')
      .groupBy(f.window('timestamp', '6 hours').alias('window'))
      .count()
      )
```

Luego vamos a configurar un writeStream que escriba a consola en modo Update esta agrupación con un trigger configurado a 3 minutos. El objetivo es que mientras el query esté corriendo, usted mueva los archivos de un directorio a otro (de lab03-data a lab03-input) uno por uno en cada uno de los intervalos de 3 minutos para simular que están siendo cargados y analice los resultados que van apareciendo en consola.

```
>>> query_c = (grouped
      .writeStream
      .outputMode('update')
      .format('console')
      .option('truncate', False)
      .trigger(processingTime='3 minutes')
      .start()
    )
```

Cuando corra el streaming deberá poder observar lo siguiente:

1. En el primer batch se podrá ver que para una de las ventanas de 6 horas hacen falta 4 registros. Estos 4 registros están repartidos en los siguientes 4 archivos y veremos como con cada batch se irá a hacer un update (o no) a la primera data generada en el primer batch.

2. Mover el archivo del segundo día al directorio lab03-input. Luego ver cómo en el resultado además de generar las cuatro filas de reporte del día 2, también hace un update a la ventana específica del día 1 para reportar que el conteo ahora se ha incrementado en 1.

3. Mover el archivo del tercer día. El resultado es el mismo que el del paso anterior, sólo que ahora actualizando el conteo a 10. ¿Por qué si configuramos el watermark a 24 horas, al procesar la data de dos días después todavía se tomó en cuenta el registro para actualizar el día 1?

4. Mover el archivo del día 4. Ahora sí podemos ver que el watermark fue aplicado, ya que el registro atrasado del día 1 ya no actualizó su respectiva ventana en la tabla resultante.

5. Mover el archivo del día 5 y completar el ejercicio viendo que con este nuevo batch tampoco se tomó en cuenta el registro atrasado.

Entrega

Para completar la entrega de este ejercicio deberá exportar el notebook en Databricks como “Source File” y subirlo al espacio indicado en Canvas.