**ACTIVIDAD 14 – BASE DE DATOS Y BIG DATA**

**RDD CONJUNTO DE DATOS DISTRIBUIDOS**

**Transformaciones:**

* **map**: Aplica una función a cada elemento del RDD, generando un nuevo RDD transformado. Es útil para modificar o transformar cada elemento de acuerdo a una lógica específica.
* **filter**: Selecciona elementos que cumplen con una condición específica definida por una función. Este método reduce el conjunto de datos solo a los elementos que pasan el filtro, lo que es útil para concentrarse en una muestra específica.
* **flatMap**: Similar a map, pero permite que cada elemento de entrada genere cero o más elementos de salida, "aplanando" los resultados en un solo RDD. Esto es útil para dividir texto en palabras individuales o procesar datos complejos.
* **union**: Combina dos RDDs en uno solo, conservando todos los elementos de ambos conjuntos de datos. Los elementos duplicados no se eliminan, por lo que pueden aparecer varias veces si estaban en ambos RDDs originales. Es útil para unir conjuntos de datos relacionados.
* **intersection**: Retorna un RDD con los elementos que son comunes a dos RDDs, eliminando duplicados en el proceso. Esta transformación es ideal para obtener la intersección de dos conjuntos de datos y ver qué elementos se repiten en ambos.
* **distinct**: Elimina los elementos duplicados dentro de un RDD, devolviendo un nuevo conjunto de datos en el que cada elemento es único. Es útil para obtener una lista sin repeticiones, como un conjunto de palabras únicas en un texto.
* **groupByKey**: Agrupa los valores de un RDD de pares clave-valor bajo la misma clave en una lista. Permite agrupar datos antes de realizar otras operaciones, como calcular promedios o totales.
* **reduceByKey**: Aplica una función de reducción a los valores de la misma clave, acumulando un solo valor por cada clave. A diferencia de groupByKey, esta transformación es más eficiente para calcular totales por clave, pues realiza la agregación en paralelo.
* **sortByKey**: Ordena los elementos en un RDD de pares clave-valor según las claves. Este método es útil para preparar reportes o presentar datos en orden específico.
* **join**: Combina dos RDDs de pares clave-valor, uniendo valores donde las claves coinciden. Es útil para asociar datos relacionados de diferentes conjuntos.
* **cogroup**: Agrupa valores de dos o más RDDs de pares clave-valor bajo las mismas claves, ideal para comparar datos de múltiples fuentes que comparten una estructura común.
* **coalesce**: Reduce el número de particiones de un RDD, lo cual optimiza el procesamiento en situaciones donde menos particiones son beneficiosas. Se utiliza al final de las operaciones, especialmente con grandes volúmenes de datos.

**Acciones:**

* **reduce**: Aplica una función que combina los elementos de un RDD secuencialmente, resultando en un solo valor. Es útil para operaciones acumulativas como suma total o cálculo de productos.
* **collect**: Recoge todos los elementos de un RDD y los devuelve como una lista en el controlador, lo cual es útil cuando se trabaja con conjuntos de datos pequeños o muestras.
* **count**: Calcula el número total de elementos en un RDD, lo que permite obtener el tamaño de un conjunto de datos o verificar resultados de filtros.
* **first**: Retorna el primer elemento de un RDD, siendo útil para inspeccionar rápidamente el inicio de un conjunto de datos.
* **take**: Recupera un número específico de elementos desde el inicio de un RDD, proporcionando una muestra de datos sin cargar todo el conjunto en la memoria.
* **saveAsTextFile**: Guarda el contenido de un RDD en un archivo de texto, lo cual es fundamental para almacenar los resultados finales en sistemas de archivos distribuidos o locales.
* **max** y **min**: Encuentran y devuelven los valores máximo y mínimo en un RDD, respectivamente. Estas acciones son útiles en análisis de datos para identificar rangos y valores extremos.
* **countByKey**: Cuenta el número de ocurrencias de cada clave en un RDD de pares clave-valor, devolviendo un diccionario con el conteo por clave. Esto es útil para categorizar o hacer análisis de frecuencias.
* **foreach**: Aplica una función a cada elemento del RDD sin devolver resultados al controlador. Ideal para ejecutar acciones sobre elementos, como registrar datos o enviar información a servicios externos.

Referencias:

<https://spark.apache.org/docs/3.5.3/index.html>

<https://keepcoding.io/blog/transformaciones-y-acciones-en-spark/>