**Spark - RDDs**

Ejemplos Transformaciones:

1. map: En PySpark, el método map() es una transformación que permite aplicar una función a cada elemento de un RDD (Resilient Distributed Dataset) de manera distribuida. La función que proporcionas se aplica de manera paralela a cada elemento del RDD, y el resultado de cada aplicación se almacena en un nuevo RDD.  
   Captura de pantalla de computadora

   Descripción generada automáticamente
2. filter: En PySpark, el método filter() es una transformación que permite filtrar los elementos de un RDD o DataFrame basándote en una condición que se aplica a cada uno de los elementos. Esta transformación devuelve un nuevo RDD o DataFrame que contiene solo los elementos que cumplen con la condición.  
   Captura de pantalla de computadora

   Descripción generada automáticamente
3. flatmap: En PySpark. flatMap es útil cuando queremos transformar cada elemento de un RDD en uno o más elementos y "aplanar" el resultado. Un caso típico es cuando queremos dividir frases en palabras.  
   Captura de pantalla de computadora

   Descripción generada automáticamente
4. union: En PySpark para combinar dos RDDs. La operación union toma dos RDDs y devuelve un nuevo RDD que contiene los elementos de ambos.  
   Captura de pantalla de computadora

   Descripción generada automáticamente
5. intersection: E n PySpark. La operación intersection devuelve un nuevo RDD que contiene solo los elementos comunes entre dos RDDs.  
   Captura de pantalla de computadora

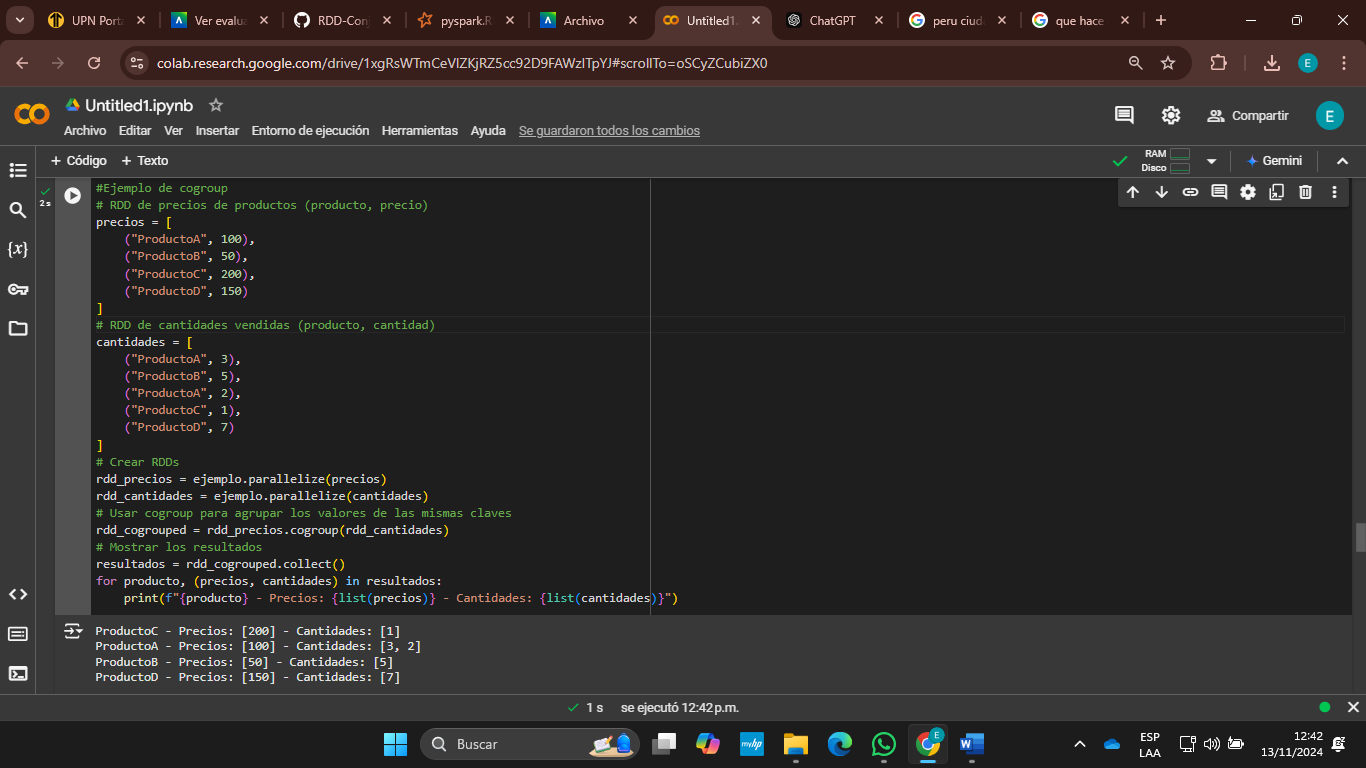
   Descripción generada automáticamente
6. distinct: E n PySpark, pero esta vez aplicándolo a un conjunto de datos más complejo, como un listado de ventas donde se repiten algunos productos y queremos obtener una lista de productos únicos.  
   Captura de pantalla de computadora

   Descripción generada automáticamente
7. groupByKey: En PySpark, la operación groupByKey se utiliza para agrupar los valores de un RDD por claves. Esto es útil cuando tienes un RDD de tuplas (clave, valor) y deseas agrupar todos los valores que comparten la misma clave.  
     
   Captura de pantalla de computadora

   Descripción generada automáticamente
8. reduceByKey: En PySpark, la operación reduceByKey se utiliza para combinar los valores de un RDD de tuplas (clave, valor) con la misma clave mediante una función de reducción. Esta operación es muy útil cuando se desea realizar un cálculo o agregación de los valores asociados con cada clave, como sumar, contar o realizar cualquier otra operación que combine los valores.  
   Captura de pantalla de computadora

   Descripción generada automáticamente
9. sortByKey: En PySpark, la operación sortByKey se utiliza para ordenar un RDD de tuplas (clave, valor) por la clave en orden ascendente o descendente. Esto es útil cuando necesitas ordenar los datos en función de las claves, y no de los valores.  
   Captura de pantalla de computadora

   Descripción generada automáticamente
10. join: E n PySpark se utiliza para combinar dos RDDs o DataFrames en función de una clave común. Cuando se aplica a un RDD de pares clave-valor, realiza una combinación de los valores asociados a la misma clave de ambos RDDs. El resultado es un nuevo RDD que contiene los pares clave-valor, donde cada clave tiene un par de valores combinados.  
    Captura de pantalla de computadora

    Descripción generada automáticamente
11. cogroup: En PySpark se utiliza para combinar dos RDDs de pares clave-valor (key-value) basándose en la clave común. A diferencia de join, que combina las claves y los valores correspondientes de ambos RDDs en un solo valor, cogroup agrupa los valores de las mismas claves en una tupla de listas. Es decir, conserva todos los valores asociados a una clave de ambos RDDs, incluso si una clave está presente solo en uno de los RDDs.  
    
12. coalesce: En PySpark se utiliza para reducir el número de particiones de un RDD o DataFrame. Es especialmente útil cuando deseas combinar varias particiones en una menor cantidad de particiones sin realizar un "shuffle" completo de los datos, lo cual puede ser más costoso en términos de rendimiento.  
      
    Captura de pantalla de computadora

    Descripción generada automáticamente

Ejemplos Acciones:

1. reduce: En PySpark, la operación reduce se utiliza para combinar los elementos de un RDD usando una función binaria (una función que toma dos argumentos y devuelve uno). Esta operación es útil cuando quieres combinar todos los elementos del RDD en un solo valor, como la suma de todos los elementos, el valor máximo, o cualquier otra operación de reducción.  
     
   Captura de pantalla de computadora

   Descripción generada automáticamente
2. collect: En PySpark, la operación collect se utiliza para recuperar los datos de un RDD o un DataFrame y traerlos al controlador (la máquina local donde está ejecutándose el código). Es una acción que provoca la ejecución de las transformaciones realizadas sobre el RDD/DataFrame y devuelve todos los elementos en una lista.  
     
   Captura de pantalla de computadora

   Descripción generada automáticamente
3. count: En PySpark, la operación count() se utiliza para contar el número de elementos en un RDD o DataFrame. Es una acción que devuelve el número total de elementos en el conjunto de datos. A diferencia de collect(), que recupera todos los datos, count() simplemente devuelve un número entero representando la cantidad de elementos.  
     
   Captura de pantalla de computadora

   Descripción generada automáticamente
4. first: En PySpark, la operación first() se utiliza para obtener el primer elemento de un RDD o DataFrame. Es una acción que devuelve el primer valor en el conjunto de datos, según el orden en que están dispuestos.  
     
   Captura de pantalla de computadora

   Descripción generada automáticamente
5. take: En PySpark, la operación take(n) se utiliza para obtener los primeros n elementos de un RDD o DataFrame. A diferencia de first(), que solo devuelve un elemento, take(n) permite obtener varios elementos de manera eficiente.  
   Captura de pantalla de computadora

   Descripción generada automáticamente
6. saveAsTextFile: La operación saveAsTextFile en PySpark se utiliza para guardar los datos de un RDD en un archivo de texto en el sistema de archivos distribuido de Hadoop (HDFS), en el sistema de archivos local o en cualquier otro sistema de almacenamiento accesible para Spark. Los datos se guardan en formato de texto, donde cada elemento del RDD se escribe en una nueva línea.  
   Captura de pantalla de computadora

   Descripción generada automáticamente
7. max,min: En PySpark, las funciones max() y min() se utilizan para obtener el valor máximo y mínimo de un conjunto de datos, respectivamente. Estas funciones se pueden aplicar tanto a un RDD como a un DataFrame.  
   Captura de pantalla de computadora

   Descripción generada automáticamente
8. countByKey: En PySpark, la función countByKey() se utiliza para contar el número de ocurrencias de cada clave en un RDD de pares clave-valor. Es una operación útil cuando necesitas contar cuántas veces aparece cada clave en un conjunto de datos.  
     
   Captura de pantalla de computadora

   Descripción generada automáticamente
9. foreach: En PySpark, el método foreach() se utiliza para aplicar una función a cada elemento de un RDD de manera local en el driver o en los nodos del clúster. A diferencia de otras funciones de transformación como map() o filter(), que devuelven un nuevo RDD, foreach() se usa para realizar acciones que no devuelven un valor, como imprimir los resultados o realizar operaciones de escritura.  
     
   Captura de pantalla de computadora

   Descripción generada automáticamente