**PREWORK**

Sesión 06

**Uso de los componentes de Spark**

Uso de los componentes de Spark

**🎯 Objetivo**

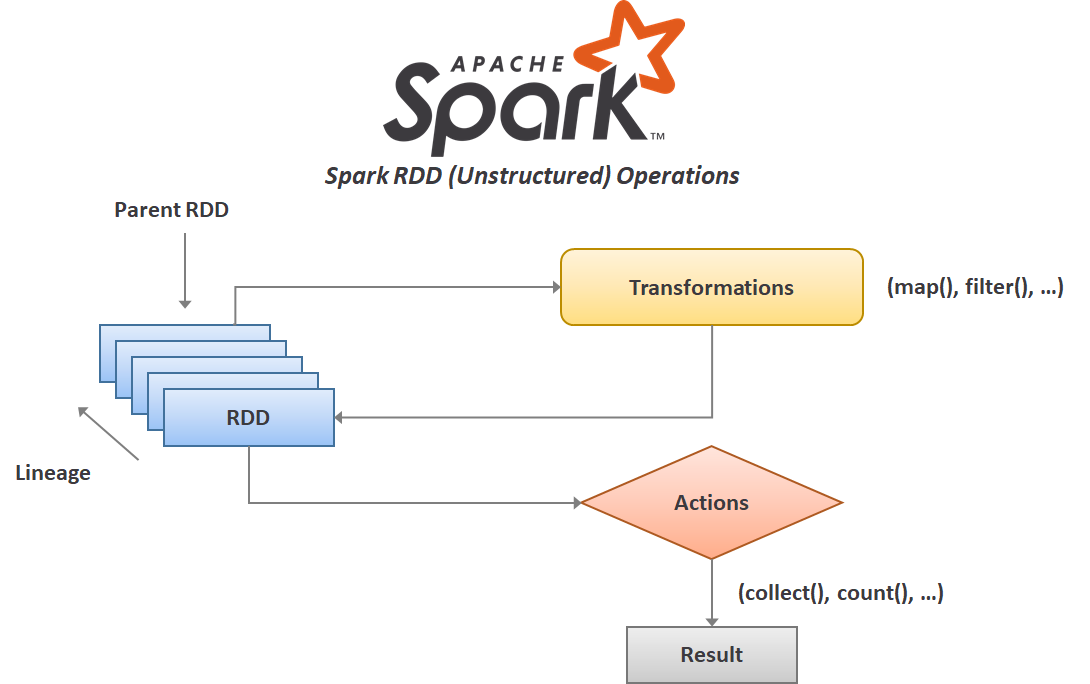
* Comprender y delimitar el alcance de RDD y su uso en Spark
* Definir y comprender la utilidad de una API en Spark con Scala o Python

### 

### 🚀 **Desarrollo**

### **Parte 1: ¿Qué es RDD?**

Cada aplicación Spark consta de un programa controlador que ejecuta la función principal del usuario y ejecuta varias operaciones paralelas en un clúster. La principal abstracción que proporciona Spark es un “conjunto de datos distribuido resistente” o mejor dicho, Resilient Distributed Dataset (RDD), lo cual es una colección de elementos particionados en los nodos del clúster que se pueden operar en paralelo. Los RDD se crean comenzando con un archivo en el sistema de archivos Hadoop (o cualquier otro sistema de archivos compatible), o incluso una colección Scala existente en el programa del controlador, y transformándola. Los usuarios también pueden pedirle a Spark que conserve un RDD en la memoria, lo que le permitirá reutilizarlo de manera eficiente en operaciones paralelas. Por último, los RDD se recuperan automáticamente de las fallas de los nodos.



Una segunda abstracción en Spark son las variables compartidas que se pueden usar en operaciones paralelas. De forma predeterminada, cuando Spark ejecuta una función en paralelo como un conjunto de tareas en diferentes nodos, envía una copia de cada variable utilizada en la función a cada tarea. A veces, es necesario compartir una variable entre las tareas o entre las tareas y el programa del controlador. Spark admite dos tipos de variables compartidas: variables de transmisión, que se pueden usar para almacenar en caché un valor en la memoria en todos los nodos, y acumuladores, que son variables a las que solo se "agregan", como contadores y sumas.

### **Parte 2: ¿Qué es una API en Spark? El caso de Python**

Una API en Spark es un método para poder “administrar” los recursos de Spark con un lenguaje tal como:

* Scala
* Python
* Java
* R

En el presente módulo usaremos Scala y Python (este último será nuestro preferido para correr scripts).

En el caso de Python, tenemos a PySpark, la cual es una interfaz para Apache Spark en Python. No solo le permite escribir aplicaciones Spark utilizando las API de Python, sino que también proporciona el shell de PySpark para analizar interactivamente sus datos en un entorno distribuido. PySpark es compatible con la mayoría de las funciones de Spark, como Spark SQL, DataFrame, Streaming, MLlib (aprendizaje automático) y Spark Core.