

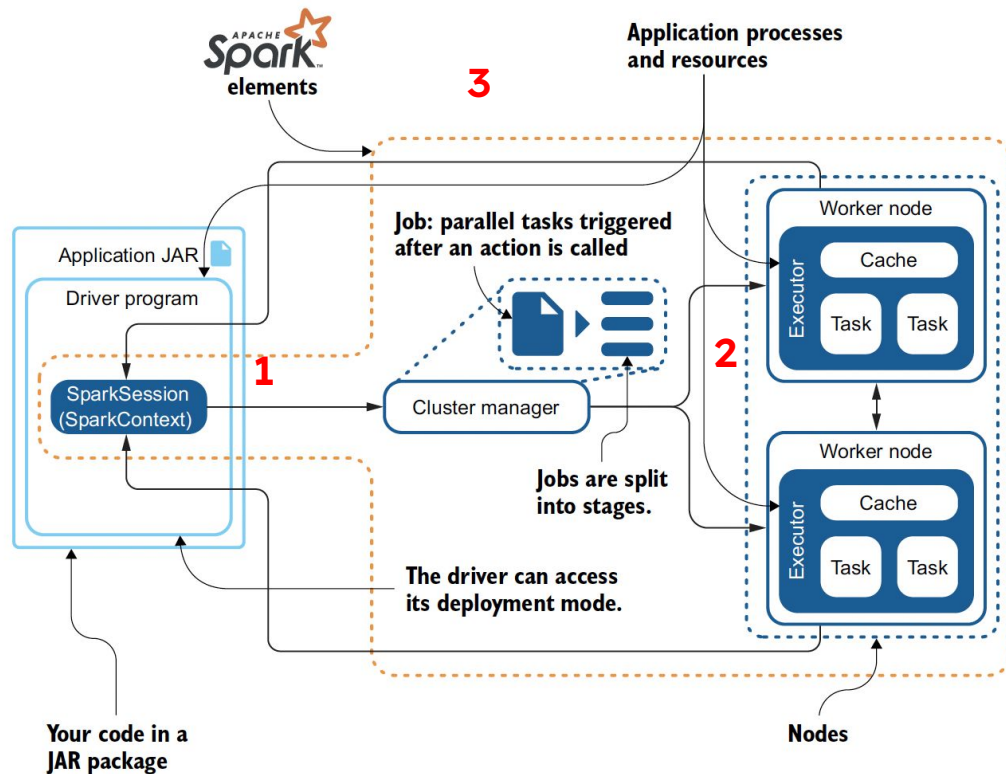


Source
Meridian

Temas de la charla

- Repaso sesión anterior
- Spark Session
- Spark context
- Spark config
- Transformaciones y Acciones en Spark
- Modo de Despliegue:
StandAlone/Client/Cluster
- Demo
 - Particiones en spark en cada nodo

Componentes de Spark I



Apache Spark components

Driver: mantener información sobre la aplicación de Spark; responder al programa o entrada de un usuario; y analizar, distribuir y programar el trabajo en los ejecutores

Ejecutor: ejecutar el código asignado por el driver y reportar el estado de la computación en ese ejecutor de vuelta al nodo driver

Configurar Spark

Spark Session:

```
SparkSession spark = SparkSession.builder()  
.appName("CSV to DB")  
.master("local")  
.getOrCreate();
```

Ofrecer una API más conveniente y unificada para trabajar con datos estructurados. Está diseñado para trabajar con DataFrames y Datasets, que proporcionan operaciones más estructuradas y optimizadas que los RDD.

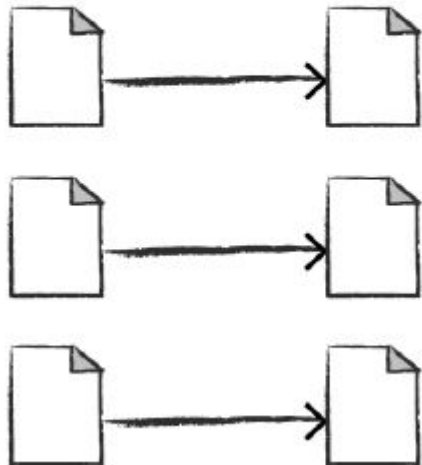
Spark Context:

```
SparkConf conf = SparkConf()  
.setAppName("CSV to DB")  
.setMaster("local[*]")
```

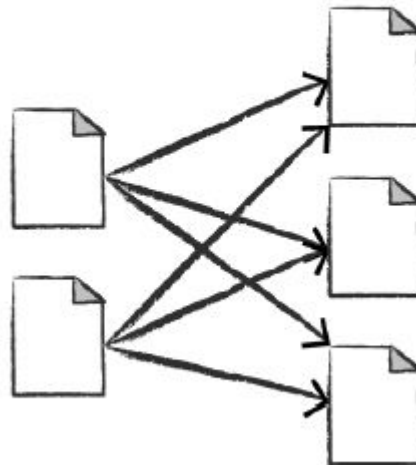
Proporciona la conexión a un clúster Spark y es responsable de coordinar y distribuir las operaciones en ese clúster. SparkContext se utiliza para la programación de RDD (Resilient Distributed Dataset) a bajo nivel.

Transformaciones y Acciones

Narrow transformations
1 to 1



Wide transformations
(shuffles) 1 to N



Transformaciones y Acciones

Las acciones son operaciones que devuelven un resultado al controlador a partir de una ejecución en un clúster

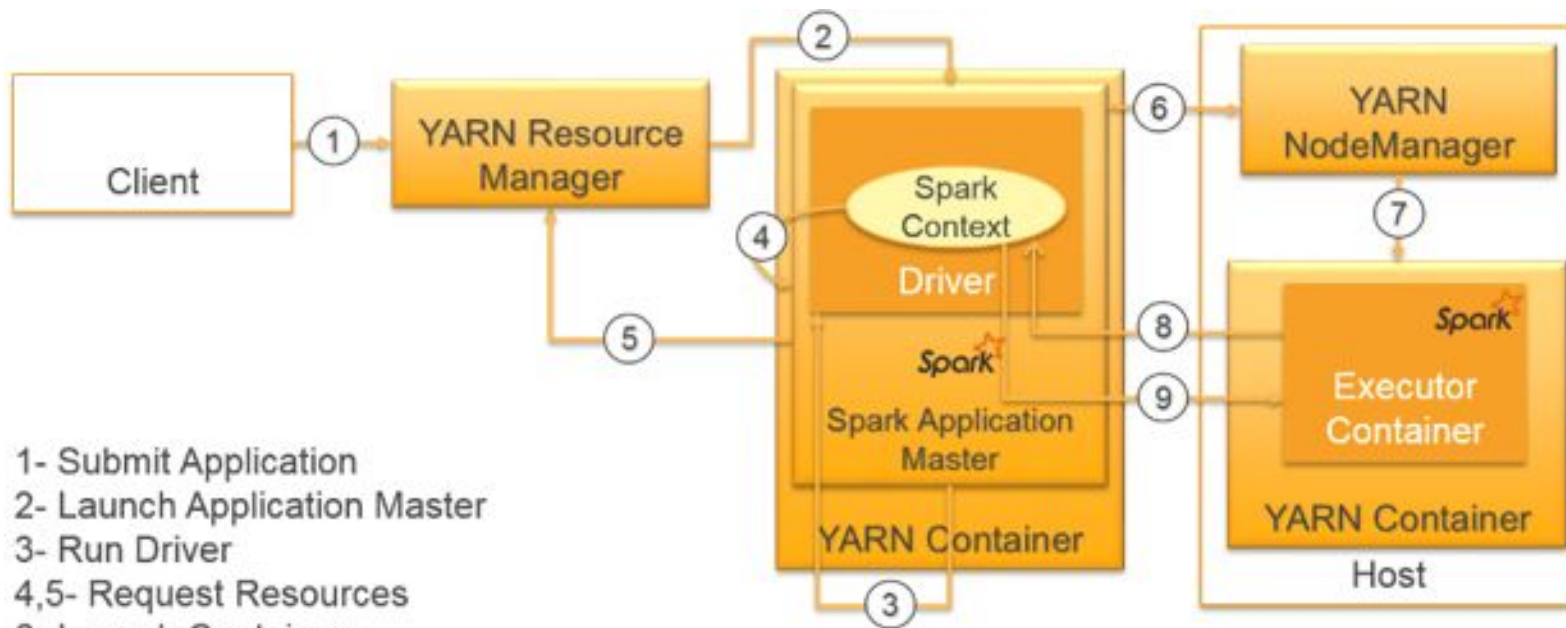
Hay tres tipos de acciones:

- Acciones para ver datos en la consola
- Acciones para recopilar datos en objetos nativos en el lenguaje correspondiente
- Acciones para escribir en fuentes de datos de salida

Lista completa de acciones en spark:

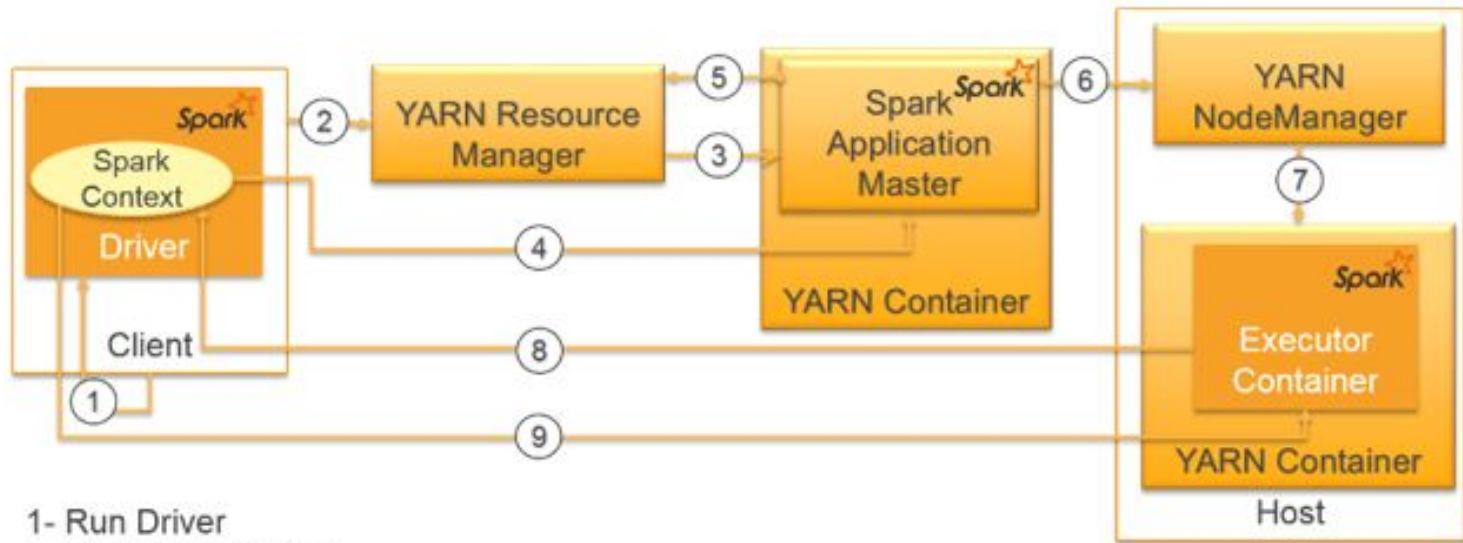
<https://spark.apache.org/docs/latest/api/scala/org/apache/spark/sql/Dataset.html>

Modos de Despliegue: Cluster



- 1- Submit Application
- 2- Launch Application Master
- 3- Run Driver
- 4,5- Request Resources
- 6- Launch Containers
- 7- Launch Spark Executors
- 8- Register with the Driver
- 9- Launch Tasks

Modos de Despliegue: Client



- 1- Run Driver
- 2- Submit Application
- 3- Launch Application Master
- 4,5- Request Resources
- 6- Launch Containers via YARN NodeManager
- 7- Launch Spark Executors
- 8- Register with the Driver
- 9- Launch Tasks

Modo Cliente

En modo cliente, los registros se generan en la máquina cliente. Es fácil de depurar y adecuado para cargas de trabajo de desarrollo.

La latencia de la red es alta

El controlador desaparece una vez que el servidor de nodo de borde se desconecta o cierra

El controlador puede quedarse sin memoria debido a que muchos usuarios utilizan el mismo nodo de borde

Modo Cluster

En modo cluster, los registros se generan en el archivo Stdout o Stderr. Es adecuado para cargas de trabajo de producción.

La latencia de la red es baja

El proceso seguirá ejecutándose en el clúster incluso si los servidores de borde están cerrados

Es poco probable que el controlador se quede sin memoria



Source Meridian

RDD vs Dataset vs Dataframe

Característica	RDD (Resilient Distributed Dataset)	DataFrame	Dataset
Nivel de API	Bajo nivel	Alto nivel	Alto nivel
Tipo de datos	Objetos JVM	Tablas estructuradas	Tablas estructuradas con tipos
Verificación de tipos	No hay verificación de tipos en tiempo de compilación	No es de tipo seguro	Verificación de tipos en tiempo de compilación (Type-safe)
Optimización	No optimizado (sin Catalyst)	Optimizado a través de Catalyst	Optimizado a través de Catalyst con ventajas adicionales
Uso de memoria	Más control sobre la memoria	Usa optimizaciones internas para reducir el uso de memoria	Usa optimizaciones internas para reducir el uso de memoria
Velocidad de ejecución	Más lento debido a la falta de optimizaciones	Más rápido que RDD debido a Catalyst	Más rápido que DataFrame debido a la generación de bytecode en JVM
Transformaciones	Amplias transformaciones disponibles	Limitado a transformaciones basadas en columnas	Limitado a transformaciones basadas en columnas, pero con tipos seguros
Acceso a datos	Orientado a objetos	Orientado a SQL/Columnas	Orientado a SQL/Columnas con tipos seguros
Casos de uso común	Procesamiento de datos complejos, cuando se necesita un control fino sobre la distribución y el paralelismo	Consultas SQL, análisis de datos, ETL	Consultas SQL, análisis de datos, ETL con tipos seguros