DIPLOMADO

Ciencia de Datos

Módulo 12

Datos masivos

Mtro. Omar Mendoza González

DGTIC

Universidad Nacional Autónoma de México

Dirección General de Cómputo y de Tecnologías de Información y Comunicación

Contenido

Apache Spark

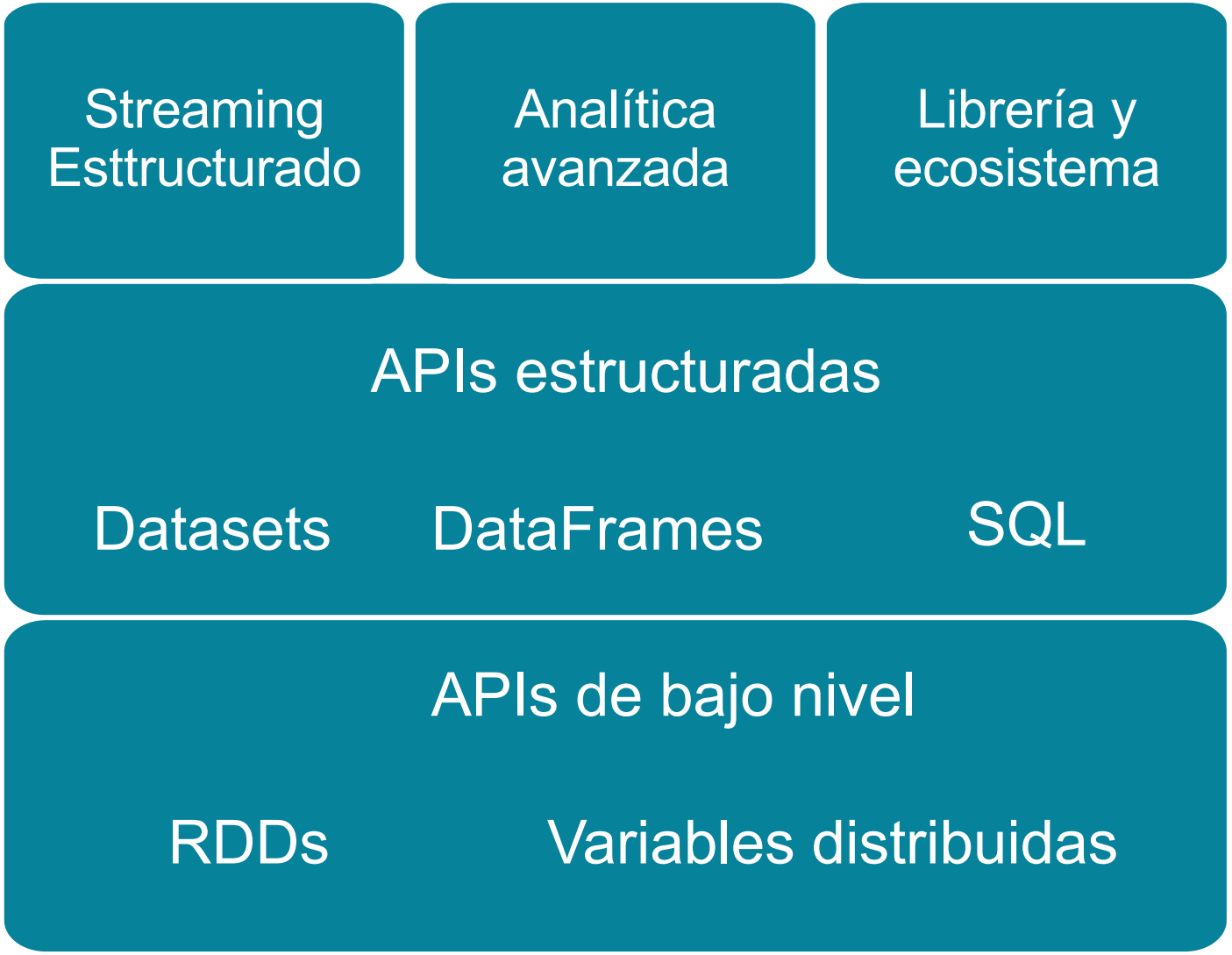
- Apache Spark es un motor de computación unificado y un conjunto de librerías para el procesamiento paralelo de datos en clusters de computadoras.

- “Apache Spark is a unified analytics engine for large-scale data processing”

Apache Spark

- Admite múltiples lenguajes de programación
  - Python
  - Java
  - Scala
  - R
  - SQL
- Incluye bibliotecas para diversas tareas que van desde SQL hasta transmisión y aprendizaje automático
- Se ejecuta desde una computadora portátil hasta un clúster de miles de servidores

Apache Spark Toolkit

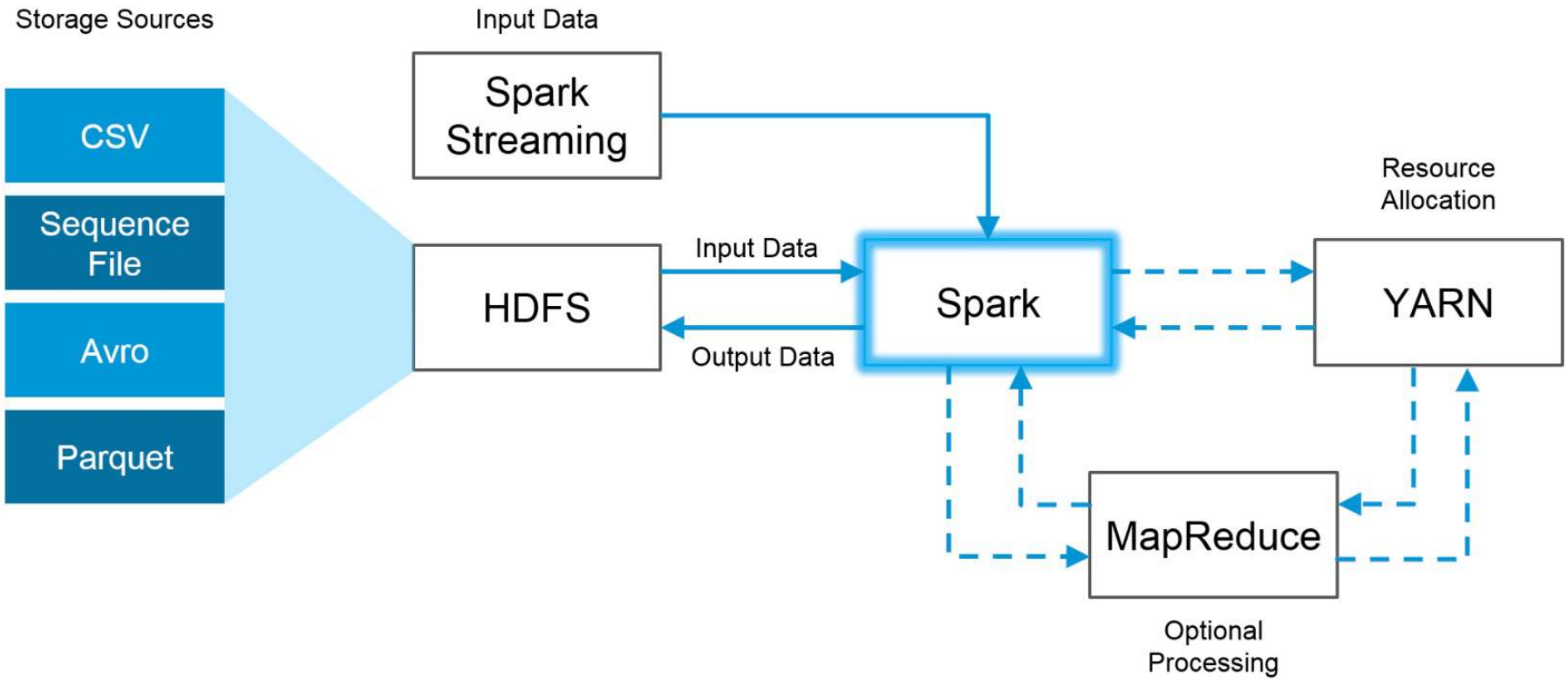


Apache Spark

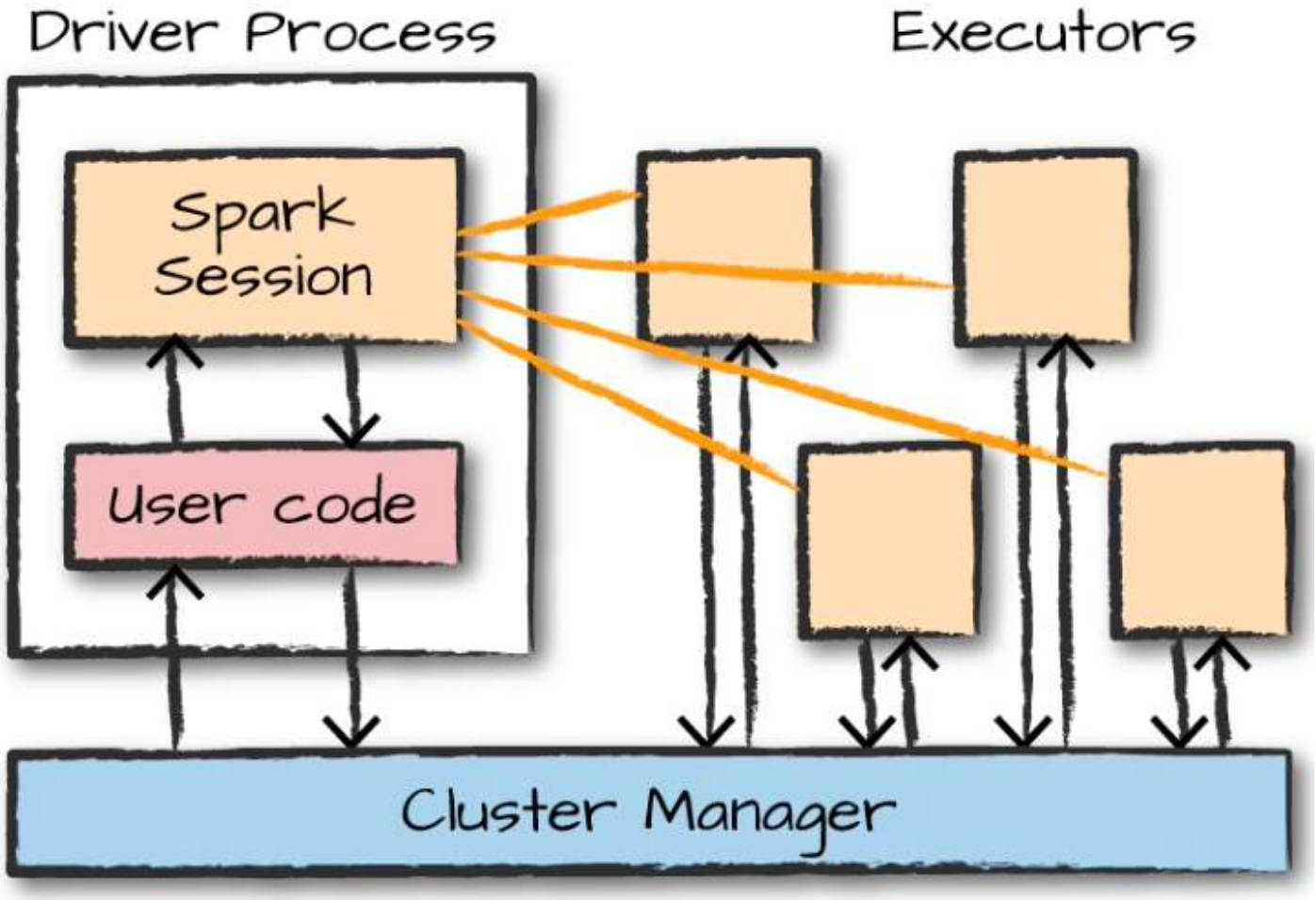
- Ofrece una plataforma unificada para escribir aplicaciones de big data (large-scale data analytics)

- Está diseñado para admitir una amplia gama de tareas de análisis de datos, que van desde la carga de datos simple y las consultas SQL hasta el aprendizaje automático.

Apache Spark



La arquitectura de una aplicación Spark



SparkContext

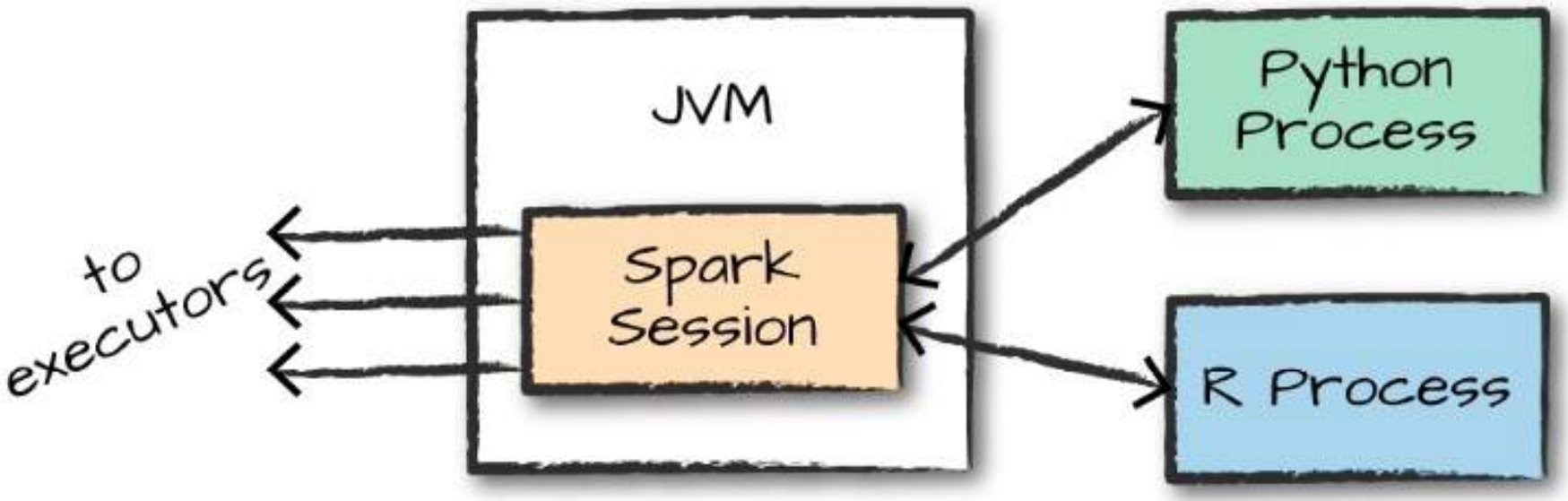
- SparkContext representa la conexión a un clúster de Spark y se puede usar para crear RDD y difundir variables en ese clúster

- Cuando crea un nuevo SparkContext , se debe establecer al menos el nombre principal de la aplicación, ya sea a través de los parámetros nombrados o mediante *conf*

SparkSession

- La aplicación Spark se controla a través de un proceso llamado ***SparkSession***.
- La instancia de SparkSession es la forma en que Spark ejecuta manipulaciones definidas por el usuario en el clúster.
- Existe una correspondencia uno a uno entre SparkSession y Spark Application.
- SparkSession puede usarse para crear *DataFrame*, registrar *DataFrame* como tablas, ejecutar SQL sobre tablas, almacenar en caché tablas y leer archivos de parquet.

SparkSession



SparkSession

```
from pyspark.sql
import SparkSession
spark =
SparkSession.builder.master("local").appName("Word
Count")\
.config("spark.some.config.option", "some-value")\
.getOrCreate()
```

Apache Spark

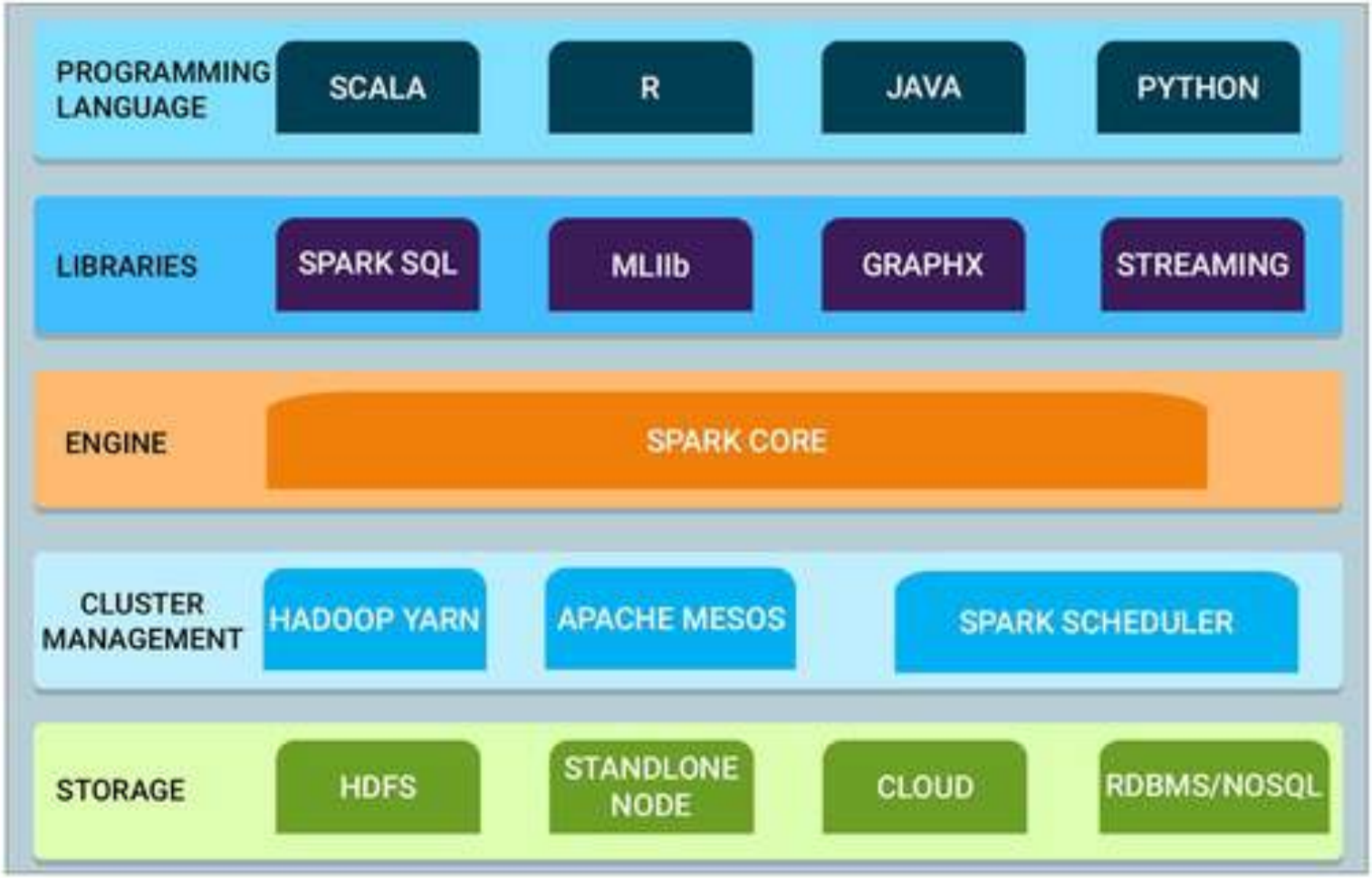
- El paradigma MapReduce es útil para procesar grandes volúmenes de datos

- Existen flujos de datos interactivos que pueden beneficierse del reuso de los datos cuando estos ya estan cargados en memoria
  - Machine learning

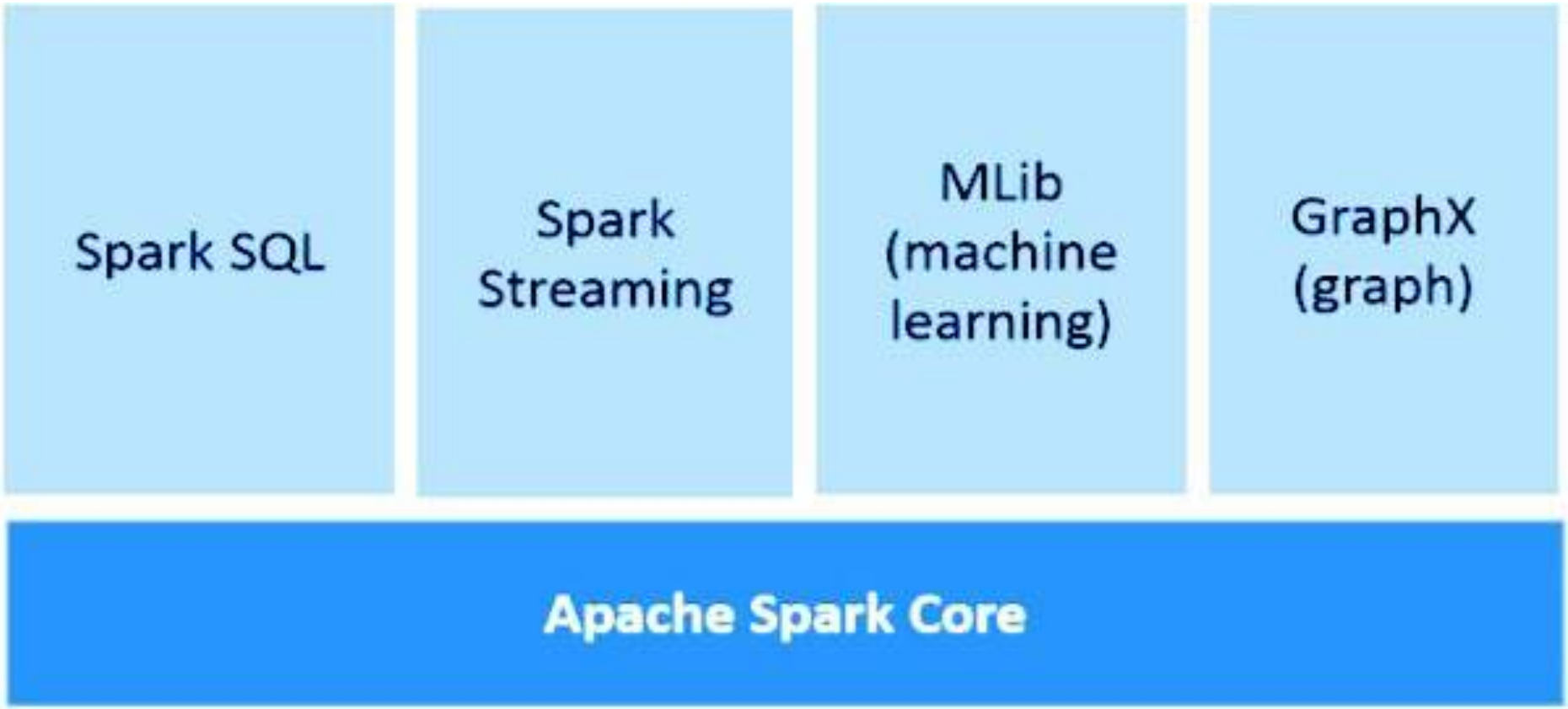
Características de Apache Spark

- Almacenamiento
  - En memoria, replicados en distintos nodos
- Multi-lenguaje
  - APIS para Java, Scala, Python y R
- Independencia de framework
  - Soporte para YARN y Mesos
- Terminal Interactiva

Compatibilidad Apache Spark



Ecosistema Apache Spark



Conceptos Spark

- Spark Shell
- Transformaciones
- Acciones
- Job
- Stages
- Tasks
- Executor
- Master
- Slave

RDD (Resilient Distributed Dataset)

- RDD es la unidad de datos central y significativa en Apache Spark.
- Pertenecen al API de bajo nivel de Spark
- Es una colección **inmutable** y **distribuida** de componentes a través de nodos de clúster y puede implementar operaciones paralelas.
- En los RDD los registros son **objetos y** puede almacenarse todo lo que se desee en estos objetos, en el formato que se desee

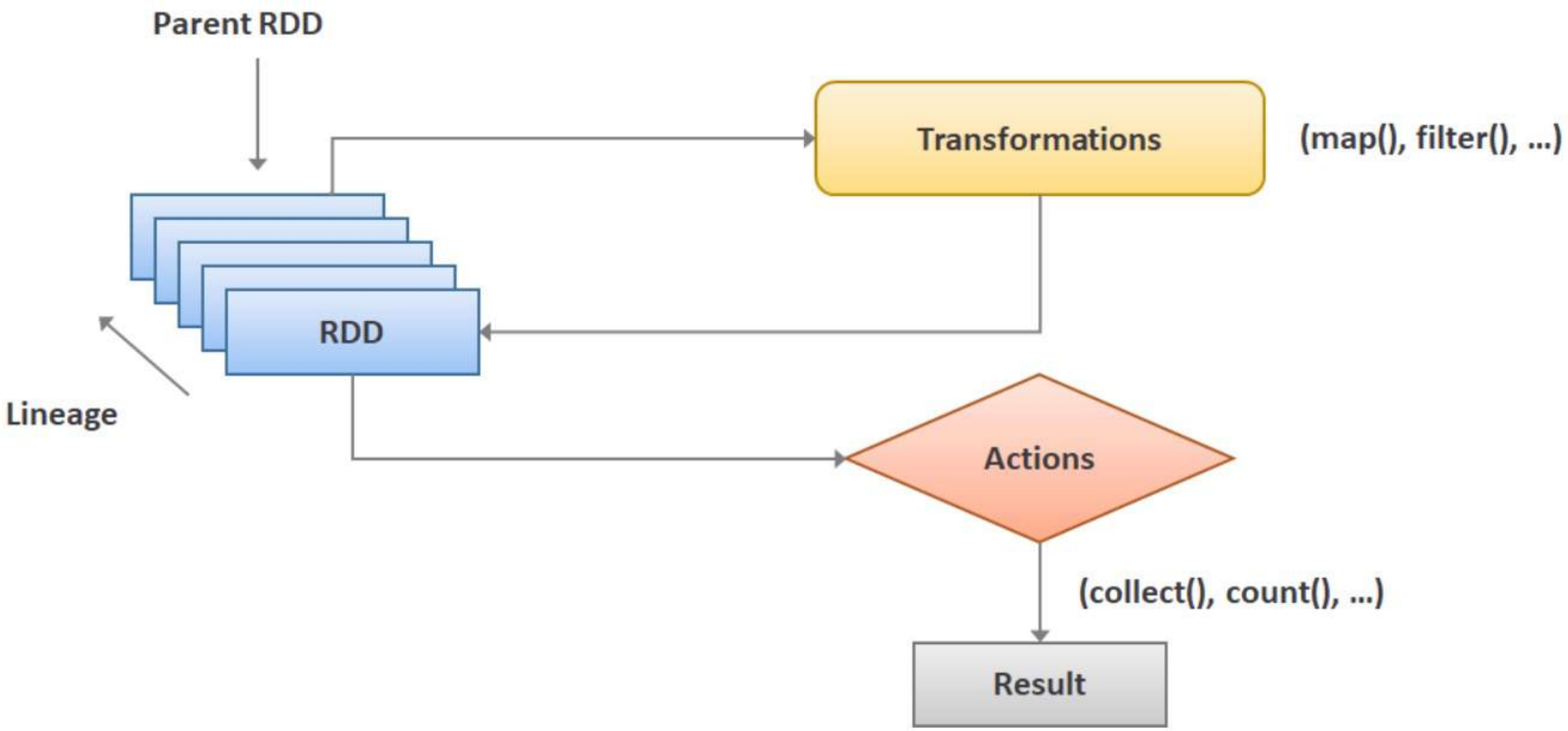
RDD

- Los RDD solo se pueden crear de dos maneras
  - Paralelizando un conjunto de datos ya existente
    - Colecciones paralelas
    - Conjuntos de datos externos (HDFS, HBase)
    - RDD existentes
  - Transformando RDD ya creados

Características de los RDD

- Inmutabilidad
- Cálculo en memoria
- Evaluación perezosa
- Tolerante a fallos
- Fraccionamiento
- Persistencia
- Operación granulada

Esquema de operaciones RDD



Transformaciones RDD

- Son los métodos que se aplican a un conjunto de datos para crear un nuevo RDD.
  - map()
  - filter()
  - distinct()
  - union()
  - intersection()
  - subtract()
  - sample()

Transformaciones RDD

- Todas las transformaciones son ***lazy***, lo que significa que no calculan sus resultados de inmediato.
- Spark recuerda las transformaciones aplicadas a algún conjunto de datos base (por ejemplo, un archivo).
- Las transformaciones solo se calculan cuando una acción requiere que se devuelva un resultado al programa controlador.
- De forma predeterminada, cada RDD transformado se puede volver a calcular cada vez que ejecuta una acción en él.
- También puede conservarse un RDD en disco, utilizando el método persist (cache).

Acciones RDD

- Son operaciones RDD que devuelven valores que no son RDD.
  - collect()
  - take()
  - count()
  - top()
  - reduce()
  - first()
  - sum()
  - aggregate()

Cuándo usar RDD

- Se requiere alguna funcionalidad que no se encuentra en un API de nivel superior
- Los datos no están estructurados
- Necesita mantener una base de código heredada escrita con RDD
- Debe realizar una manipulación personalizada de variables compartidas.
- No importa imponer un esquema
- No importa renunciar a algunos beneficios de optimización y rendimiento

DAG (Direct Acyclic Graph)

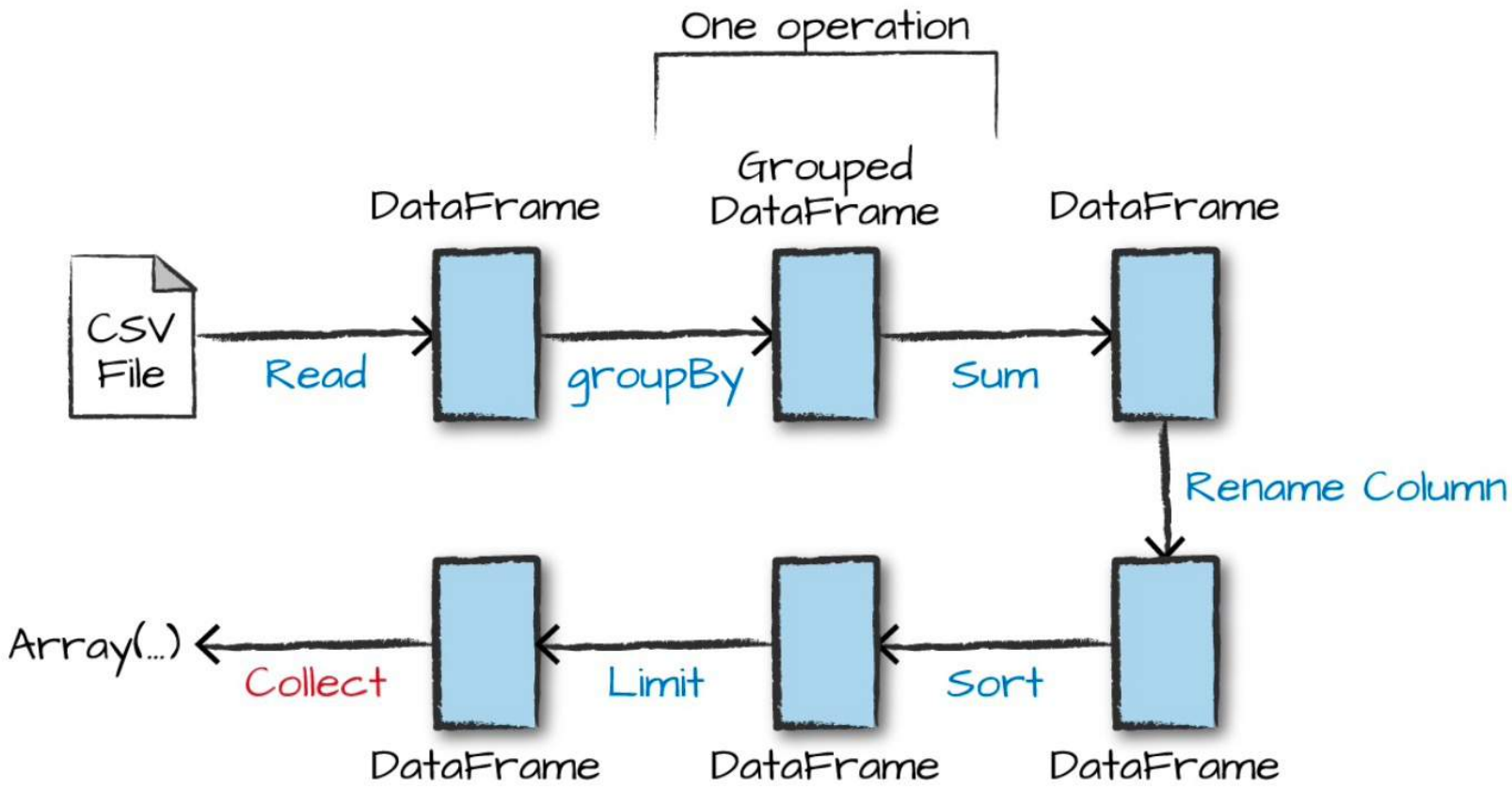
- Cada tarea de Spark crea un DAG de etapas de trabajo para que se ejecuten en un determinado cluster.

- En comparación con MapReduce, el cual crea un DAG con dos estados predefinidos (Map y Reduce), los grafos DAG creados por Spark pueden tener cualquier número de etapas.

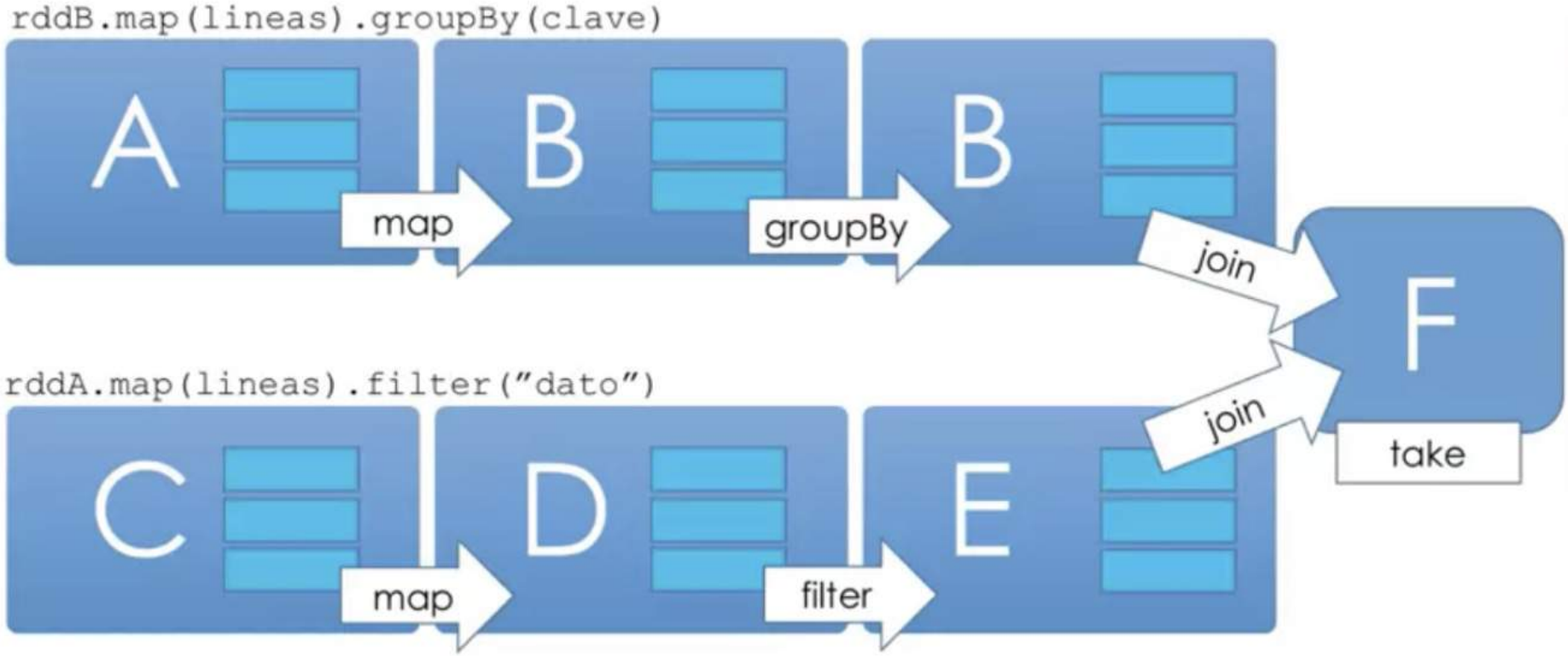
- Spark con DAG es más rápido que MapReduce por el hecho de que no tiene que escribir en disco los resultados obtenidos en las etapas intermedias del grafo.

DAG (Direct Acyclic Graph)

- Todo el flujo de transformación de DataFrame



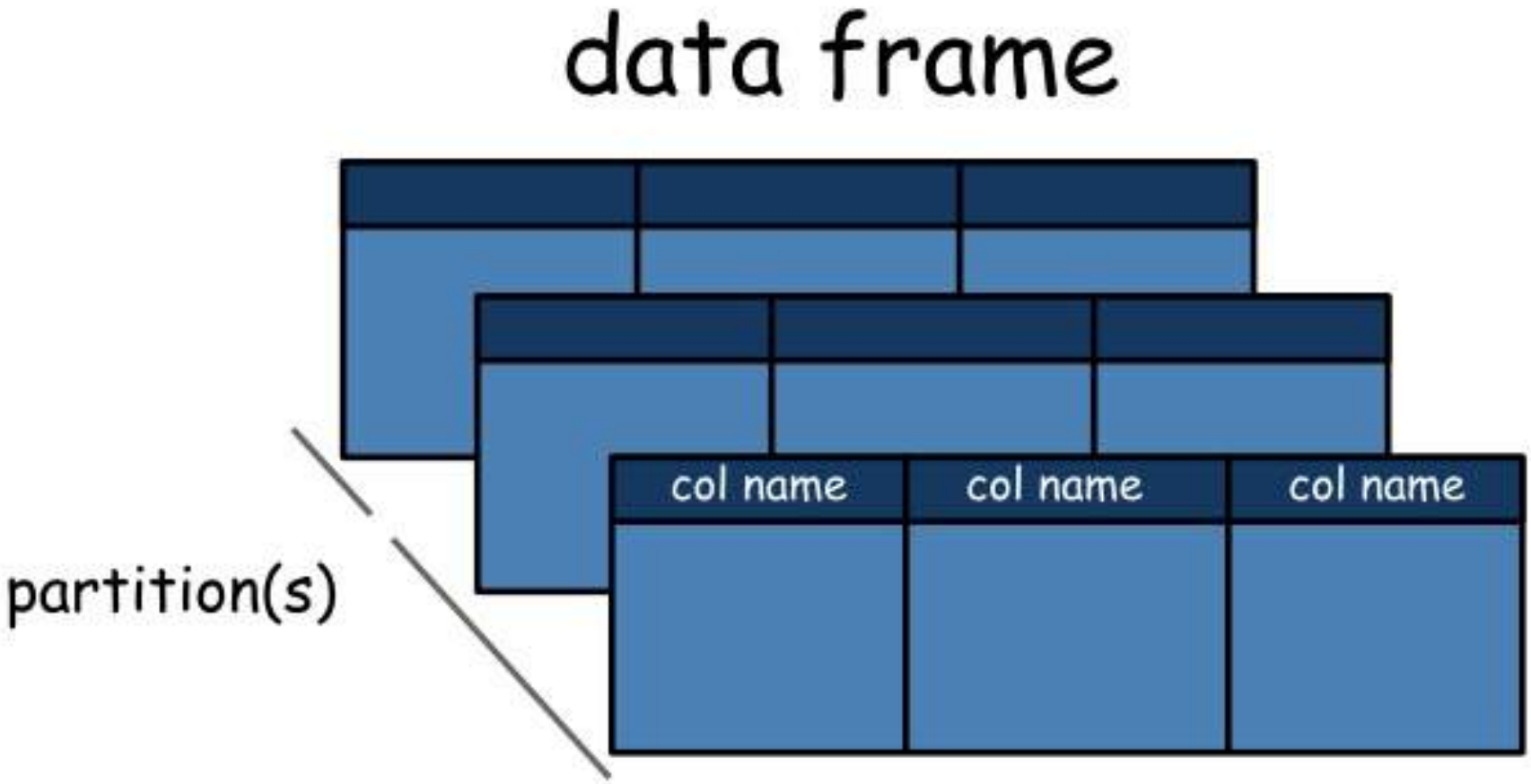
DAG (Direct Acyclic Graph)



DataFrame

- DataFrame es la API estructurada más común y simplemente representa una tabla de datos con filas y columnas.
- La lista que define las columnas y los tipos dentro de esas columnas se llama esquema.
- El DataFrame puede distribuirse en un cluster.

DataFrame



DataFrame

- Se pueden construir a partir de una amplia gama de fuentes, como archivos de datos estructurados, tablas en Hive, bases de datos externas o RDD existentes.

Particiones

- Los datos de cada partición residen en una sola máquina.
- Spark crea una tarea para cada partición.
- Las operaciones de Spark Shuffle mueven los datos de una partición a otras particiones.
- El particionamiento es una operación costosa ya que crea una mezcla de datos

Particiones

- Para permitir que cada ejecutor realice un trabajo en paralelo, Spark divide los datos en fragmentos llamados particiones.
- Una partición es una colección de filas que se ubican en una máquina física en un clúster.
- Normalmente, Spark intenta establecer la cantidad de particiones automáticamente en función de su clúster

Contacto

Omar Mendoza González

Profesor de carrera ICO FES Aragón

omarmendoza564@aragon.unam.mx

Referencias

- **Corea, Francesco**, **An Introduction to data : everything you need to know about AI, Big data and data science / Francesco Corea -- Cham, Switzerland :** Springer, [2019].-- xv, 131 páginas : ilustraciones (Studies in Big data, 2197-6503 ; 50)

- **Casas Roma, Jordi**, **Big data : análisis de datos en entornos masivos / Jordi Casas Roma, Jordi Nin Guerrero, Francesc Julbe López -- Barcelona :** Editorial UOC, 2019 287 páginas : ilustraciones (Tecnología ; 623).

- **Caballero, Rafael, Big data con Python recolección, almacenamiento y proceso /** Rafael Caballero Adrián Riesco Enrique Martín: Universidad Complutense de Madrid Editorial AlfaOmega, 2019 282 páginas

- **Rioux, Jonathan, Data Analysis with Python and PySpark** / Jonathan Rioux: Editorial Manning Publications, 2020 259 páginas

- **Singh, Pramod, Machine Learning with PySpark: With Natural Language Processing and Recommender Systems** / Pramod Singh: Editorial Apress, 2019 233 páginas