GCDSTE
ENSA MARRAKECH

المدرسة الوطنية للعلوم التطبيقية - مراكش
ÉCOLE NATIONALE DES SCIENCES APPLIQUÉES - MARRAKECH
جامعة القاضي عياض
UNIVERSITÉ CADI AYYAD

Apache Spark : Anatomie d'un Moteur de Calcul Distribué

Une Plongée Visuelle au Cœur de l'Exécution des Données

Réalisé par:

- ***Yahya Bennani***
- ***Sayf Eddine Laamri***
- ***Oussama Bagy***
- ***Iliass Chbani***
- ***Saad Amar***
- ***Ismail Amsou***

Encadré par: ***Pr. Bekkari Aissam***

Le Défi du Big Data : Pourquoi la Vitesse est Devenue Cruciale

Le Contexte : Apache Spark est un framework open source conçu pour le traitement de données à grande échelle.

hadoop MapReduce

APACHE Spark

L'Objectif Initial : Surmonter les limitations de MapReduce, notamment sa lenteur due aux lectures/écritures disque constantes, particulièrement pour les traitements itératifs et interactifs.

La Promesse de Spark : Un traitement unifié et rapide pour le batch, le streaming, le machine learning et les graphes.

Le Secret de la Vitesse : Le Calcul en Mémoire (In-Memory)

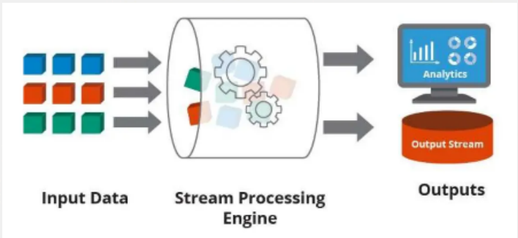
Le Modèle MapReduce

Traitement par lot (batch processing)

Chaque opération lit et écrit sur le système de fichiers distribué (HDFS).
La latence du disque est le goulot d'étranglement.

Le Modèle Spark

Traitement par volé (stream processing)



Les données intermédiaires sont 'cachées' en mémoire vive. L'accès est des centaines de fois plus rapide, ce qui est idéal pour les algorithmes itératifs.

En Bref : Les Points Clés à Retenir sur Spark

Avantages

Vitesse : Grâce au traitement in-memory.

Polyvalence : Une plateforme unifiée (SQL, Streaming, ML, Graphes).

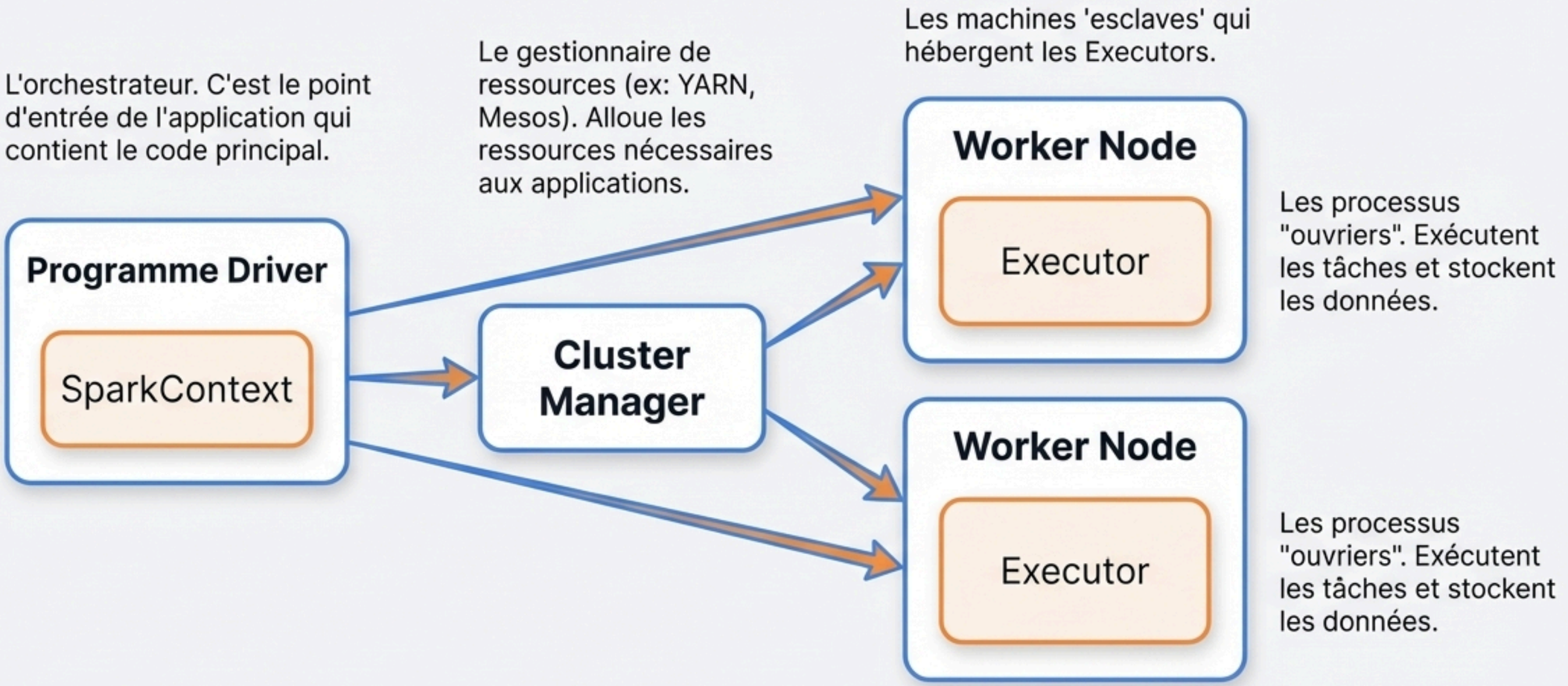
Flexibilité : APIs disponibles en Scala, Java, Python, et R.

Résilience : Tolérance aux pannes intégrée via le lignage des RDDs.

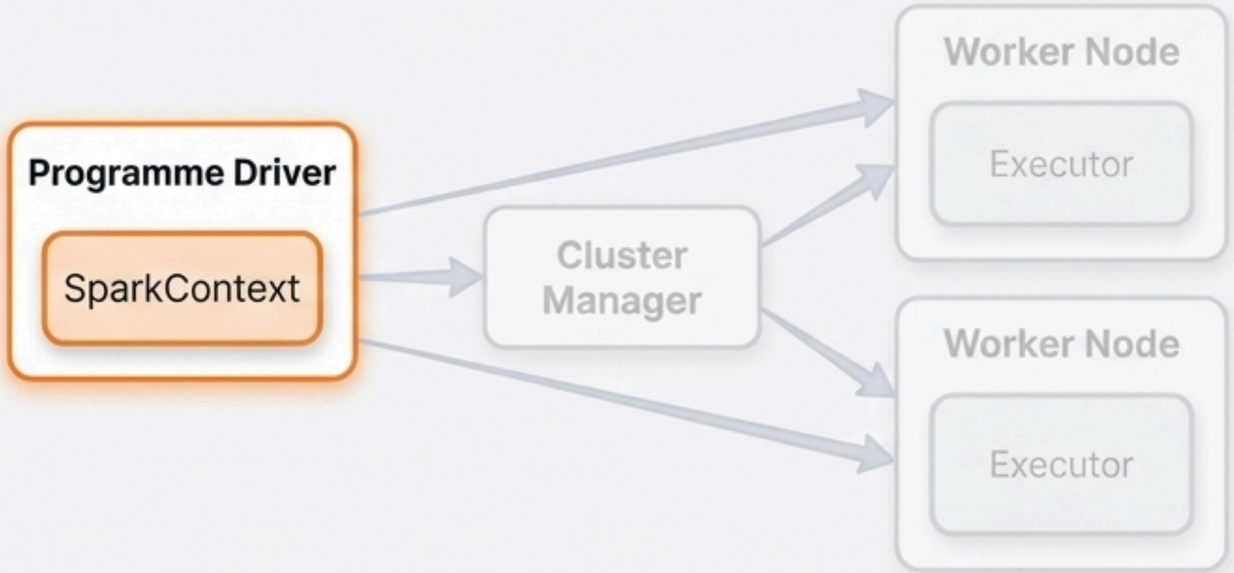
Contraintes

Coût en Mémoire RAM : La performance a un prix. Le traitement en mémoire nécessite des ressources matérielles importantes.

L'Architecture Spark : Les Acteurs d'un Cluster



Zoom sur le Driver : Le Cerveau de l'Application



- Contient la fonction `**main()**` de l'application.
- Initialise le **SparkContext** (le cœur de la connexion au cluster).
- Transforme le code utilisateur en un plan d'exécution logique : le **DAG (Directed Acyclic Graph)**.
- **Négocie** les ressources (Executors) avec le Cluster Manager.
- **Envoie** les tâches aux Executors pour exécution.

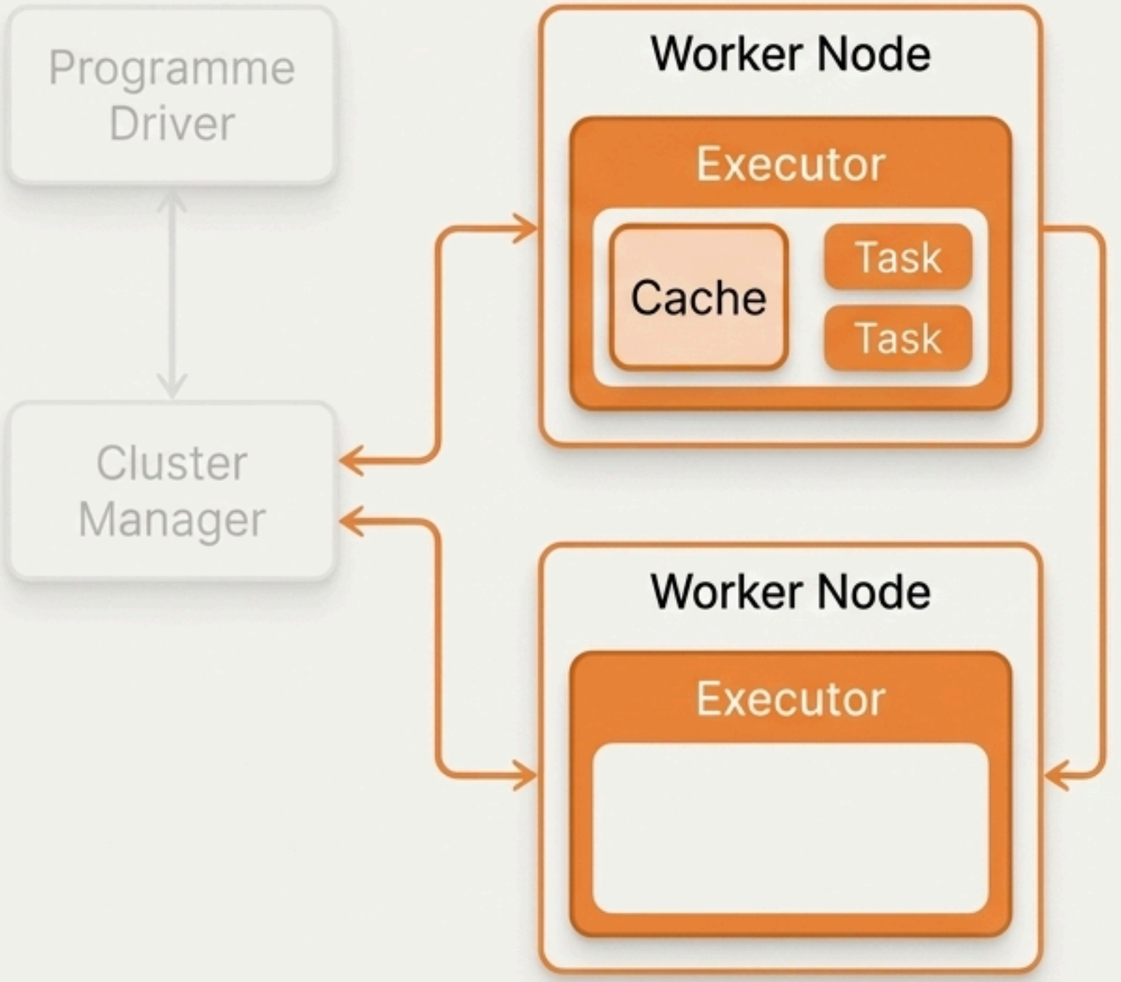
Zoom sur les Workers et Executors : La Force de Calcul

Worker Node : Une machine (physique, VM, conteneur) du cluster.

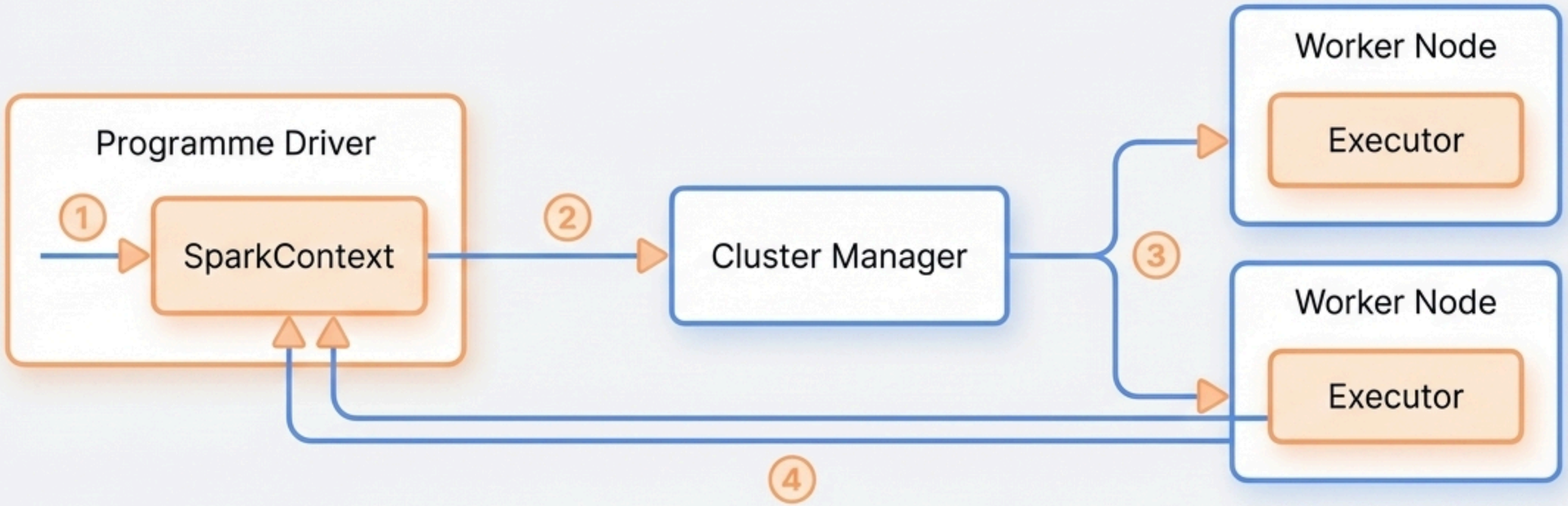
Executor : Un processus JVM lancé sur un Worker Node, dédié à une application Spark. Spark.

Fonctions de l'Executor :

1. Exécuter les tâches (**Task**) assignées par le Driver.
2. Stocker les partitions de données en mémoire ou sur disque (**Cache**).
3. Renvoyer les résultats au Driver.



Anatomie d'un Job Spark : Soumission et Planification (1/2)



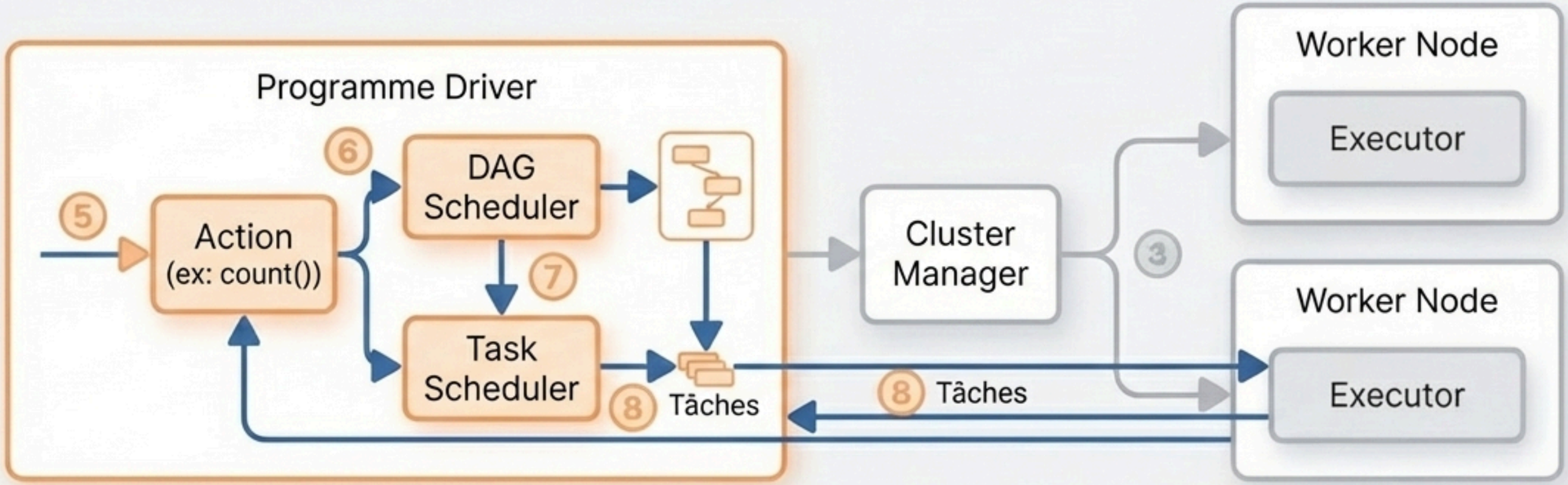
1. **Soumission du Code :** Le Driver exécute le code utilisateur et initialise le SparkContext.

2. **Requête de Ressources :** Le SparkContext communique avec le Cluster Manager pour demander des Executors.

3. **Allocation des Ressources :** Le Cluster Manager lance les processus Executor sur les Worker Nodes disponibles.

4. **Enregistrement des Executors :** Les Executors, une fois lancés, s'enregistrent directement auprès du Driver. La communication est établie.

Anatomie d'un Job Spark : Exécution et Résultat (2/2)



5. **Création du Job** : Une 'Action' dans le code (ex: `count()`, `collect()`) déclenche la création d'un Job.

6. **Planification DAG** : Le DAG Scheduler convertit le plan logique en un plan physique de 'Stages'.

7. **Ordonnancement des Tâches** : Le Task Scheduler prend les Stages et crée des tâches concrètes qu'il peut envoyer aux Executors.

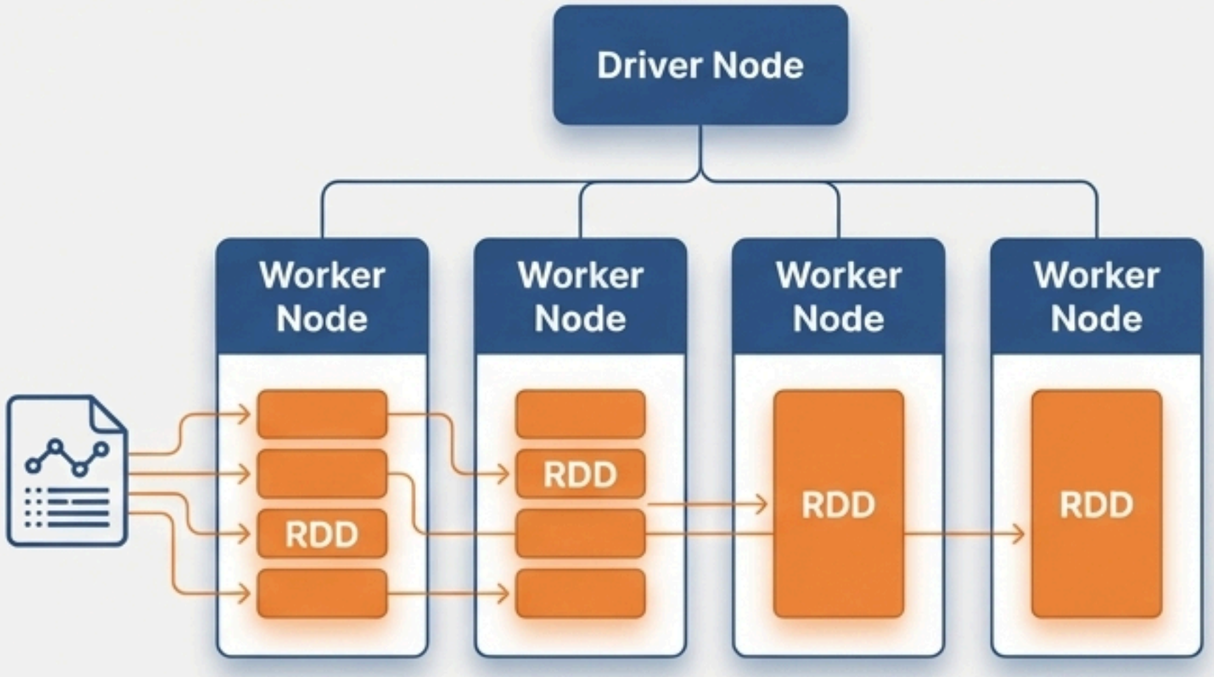
8. **Distribution et Exécution** : Les tâches sont envoyées aux Executors. Ils les exécutent sur leurs partitions de données et renvoient les résultats au Driver.

Le Concept Fondamental : RDD (Resilient Distributed Dataset)

Resilient : Tolérant aux pannes. Spark conserve l'historique des transformations (le 'lineage') pour reconstruire toute partition perdue.

Distributed : La collection de données est partitionnée et répartie sur les différents nœuds du cluster.

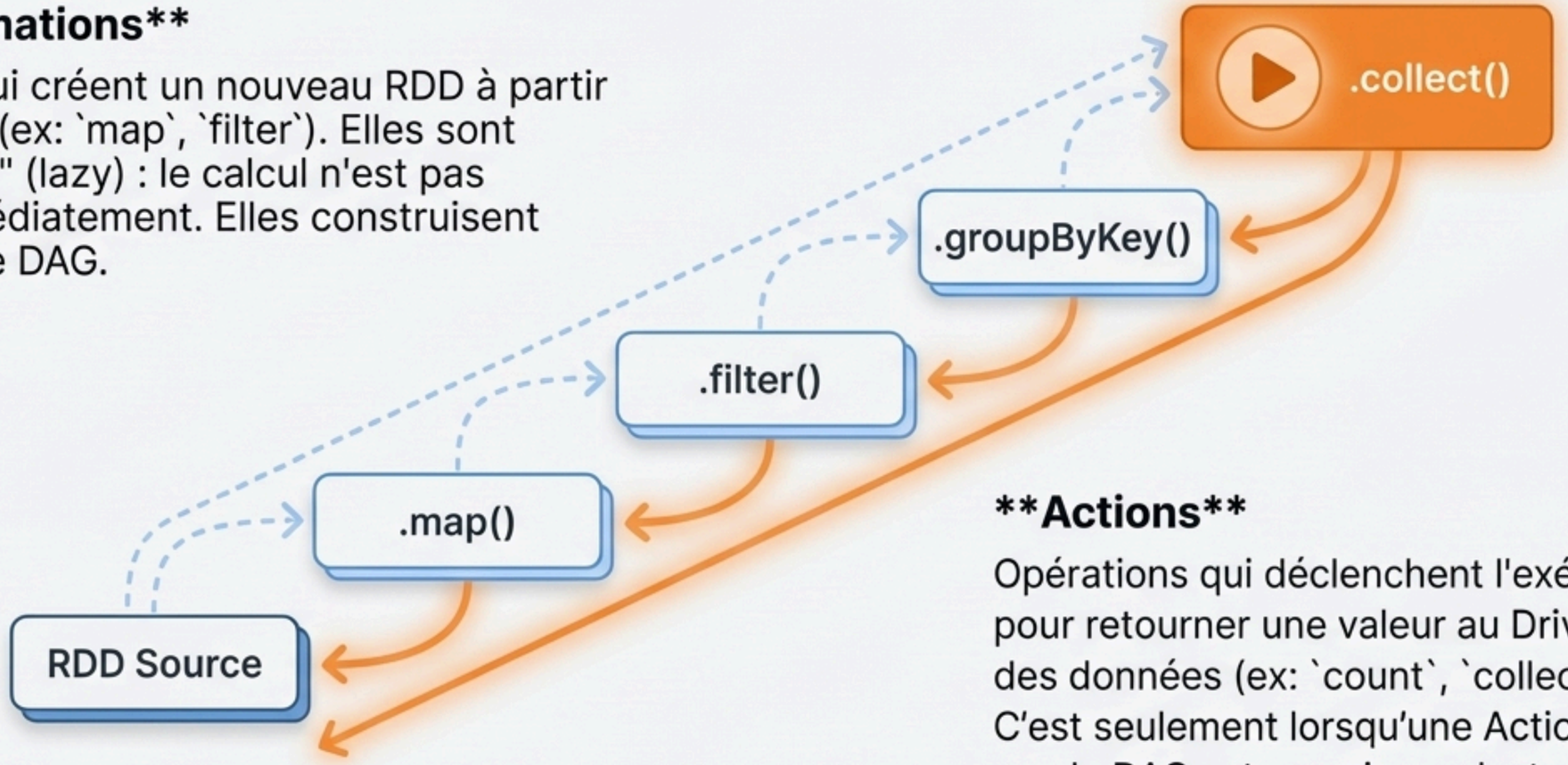
Dataset : Une collection d'objets immuables (en lecture seule) qui peuvent être traités en parallèle.



Le Déclencheur : Transformations (Lazy) et Actions (Eager)

Transformations

Opérations qui créent un nouveau RDD à partir d'un existant (ex: `map`, `filter`). Elles sont "paresseuses" (lazy) : le calcul n'est pas exécuté immédiatement. Elles construisent simplement le DAG.

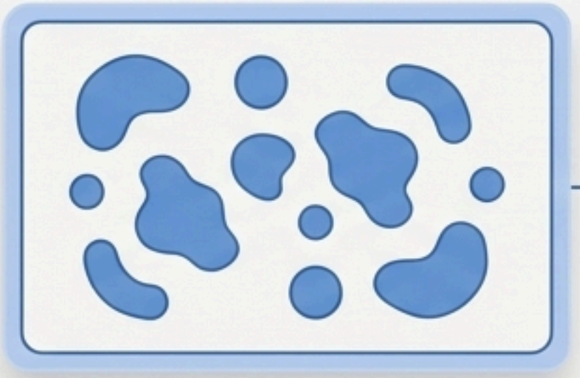


Actions

Opérations qui déclenchent l'exécution d'un job pour retourner une valeur au Driver ou écrire des données (ex: `count`, `collect`, `save`). C'est seulement lorsqu'une Action est appelée que le DAG est soumis au cluster.

L'Évolution : Des RDDs aux DataFrames et Datasets

RDD : Boîte Noire



Puissant mais manque d'information sur la structure des données. Spark ne peut pas optimiser le traitement.

DataFrame : Structuré

ID	Name	Age
1	Allen	23
2	Bobby	35
...	...	...

Une abstraction qui organise les données en colonnes nommées, comme une table de base de données.

DataSet : Typé et Structuré

ID: long	Name: string	Age: int
1	Allen	23
2	Bobby	35
...	...	...

****Avantages Clés**** : Permet des optimisations massives via le moteur Catalyst et une intégration parfaite avec Spark SQL. Il est aujourd'hui préférable d'utiliser ces API de haut niveau.

Spark en Action : Des Cas d’Usage qui Transforment les Industries

Pipelines ETL et Business Intelligence

Traitement et transformation de grands volumes de données pour les data warehouses.

Analyse de Données en Temps Réel

Détection de fraude, surveillance de logs, analyse de flux IoT.

Machine Learning à Grande Échelle

Systèmes de recommandation (e-commerce, médias), classification, clustering.

Analyse Génomique et Recherche Scientifique

Traitement de vastes ensembles de données de recherche.

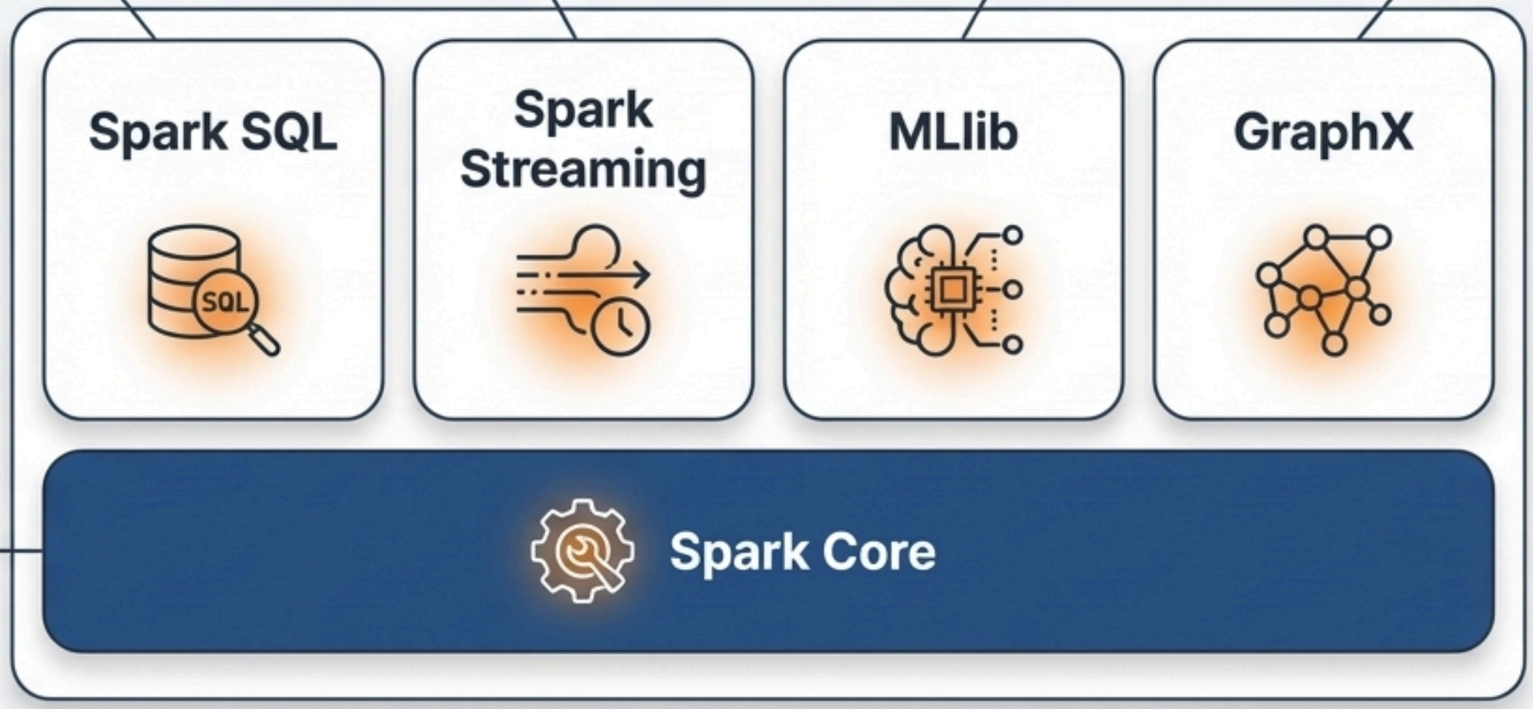
Plus qu’un Moteur : Un Écosysteme Complet et Intégré

Pour interroger les données structurées via SQL ou DataFrames.

Pour l'analyse de flux de données en quasi-temps réel.

Bibliothèque intégrée de Machine Learning scalable.

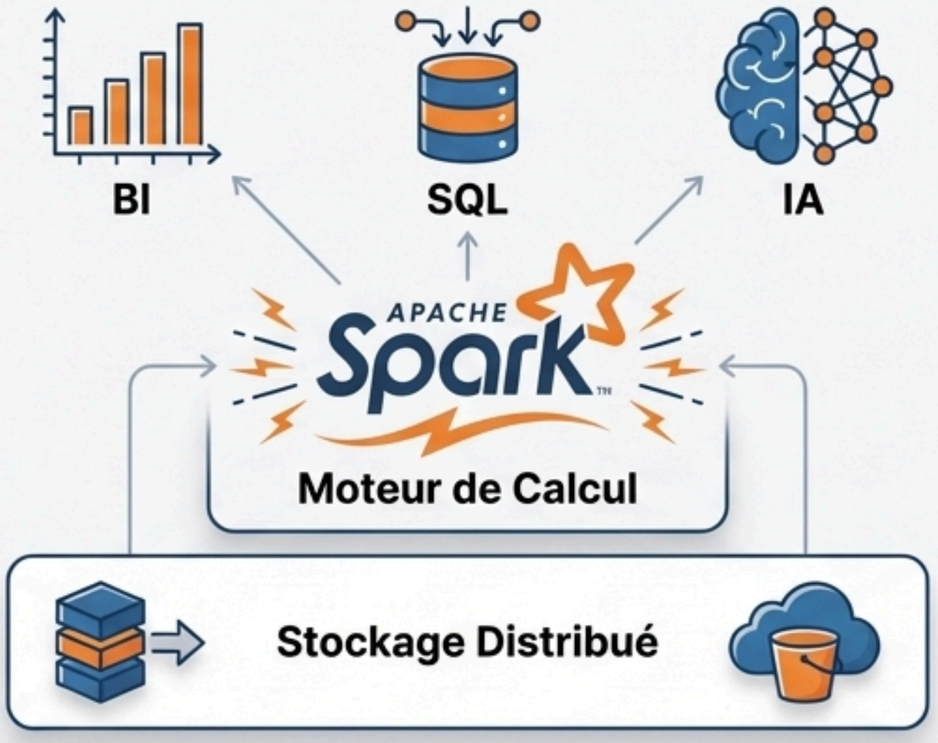
Pour le traitement distribué des graphes.

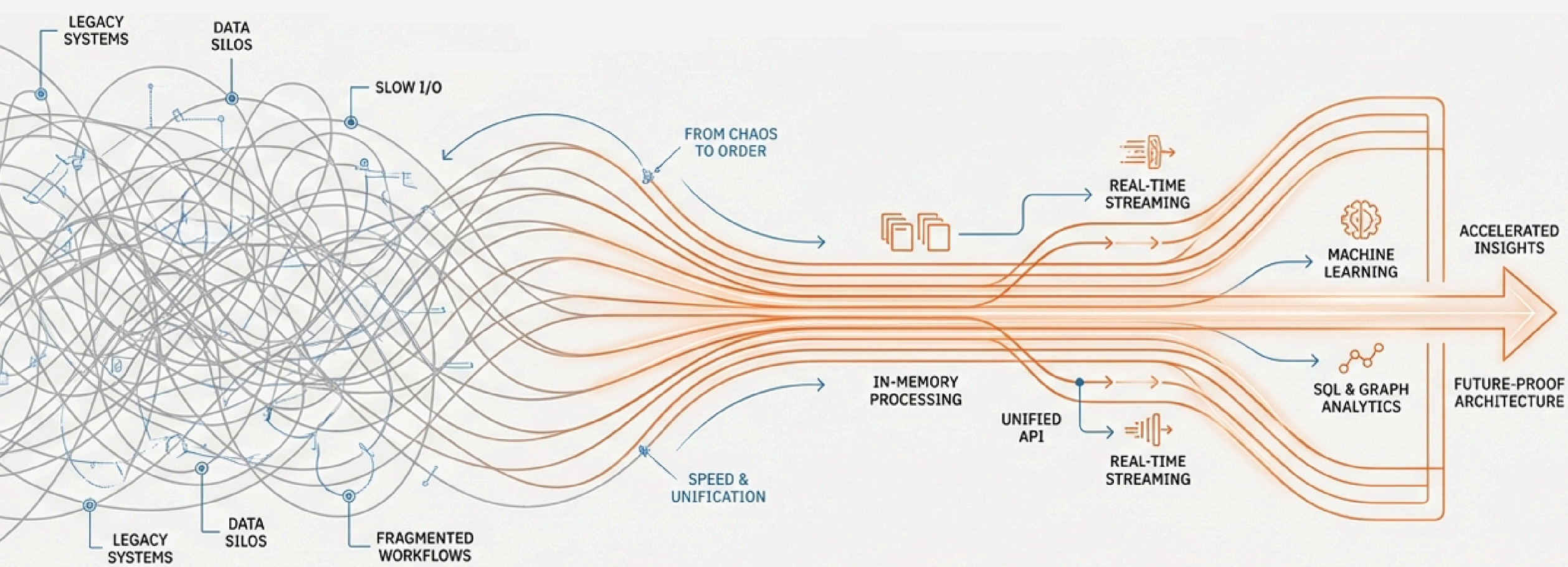


Le cœur du réacteur. Gère la distribution des tâches et les fonctionnalités de base.

Spark : Le Moteur Standard pour l’Interaction avec les Données

" En combinant le stockage distribué comme HDFS avec la puissance de calcul en mémoire de Spark, nous avons la pile technologique fondamentale qui alimente l'analyse de données moderne, de la BI à l'intelligence artificielle. "





merci pour votre attention