Chapitre Framework
Spark

Mr DIATTARA Ibrahima

Sommaire

Qu'est-ce que Spark

Spark est un Framework pour les calculs distribués (répartis ou partagés), on repartie le traitement sur plusieurs microprocesseur de différentes machines

Spark est en développé en scala

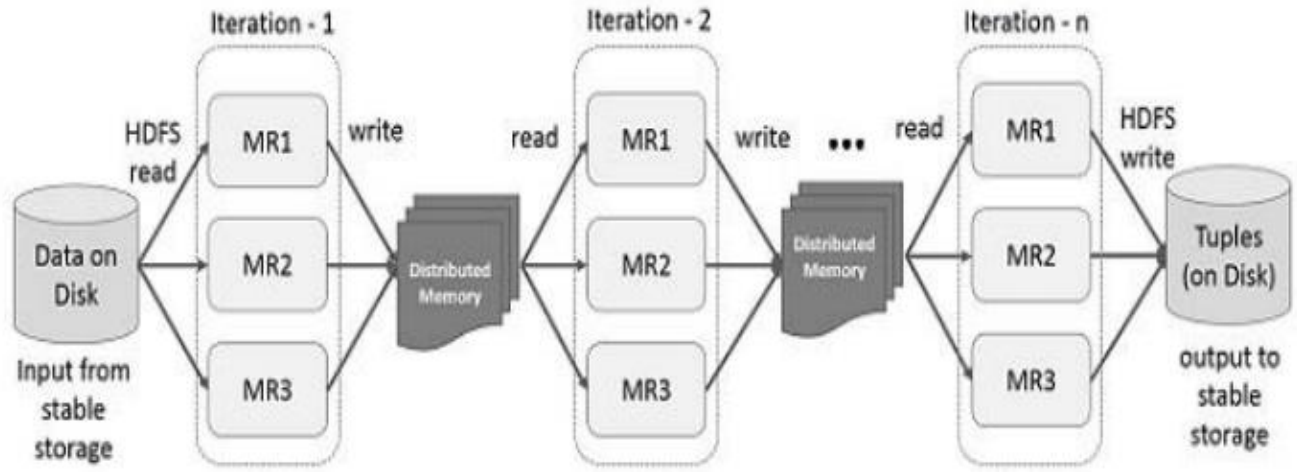
Construit pour effectuer des analyses sophistiquées et conçu pour la rapidité et la facilité d’utilisation

Mapreduce Hadoop vs Spark

Traitement des données en mémoire :

•**MapReduce** : MapReduce traite les données de manière séquentielle et stocke les résultats intermédiaires sur le disque après chaque étape du traitement. Cela peut entraîner des E/S disque coûteuses, ce qui peut ralentir les performances.

•**Spark** : Spark effectue le traitement en mémoire autant que possible, minimisant ainsi les E/S disque. Il utilise une structure de données résiliente distribuée (RDD) pour stocker les données en mémoire, ce qui accélère considérablement les opérations.

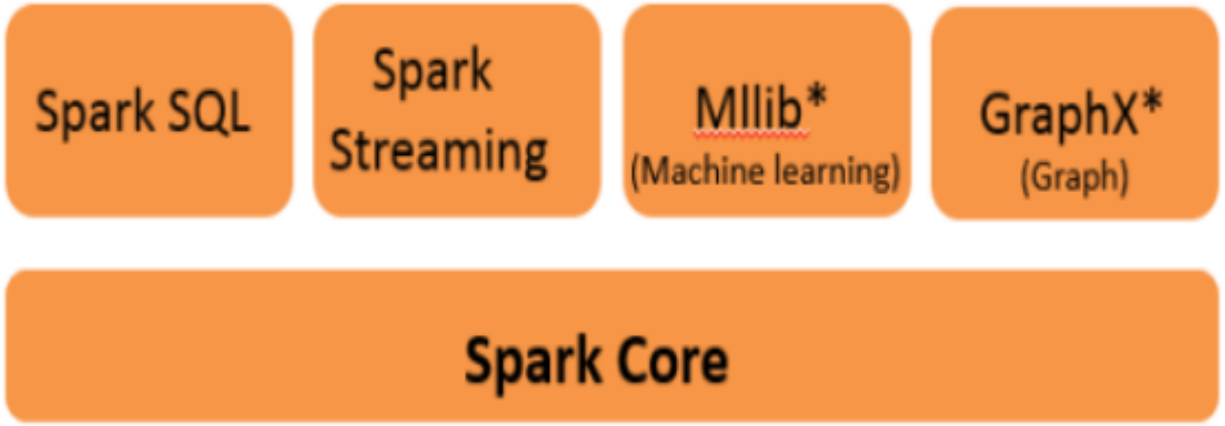


Fonctionnalités

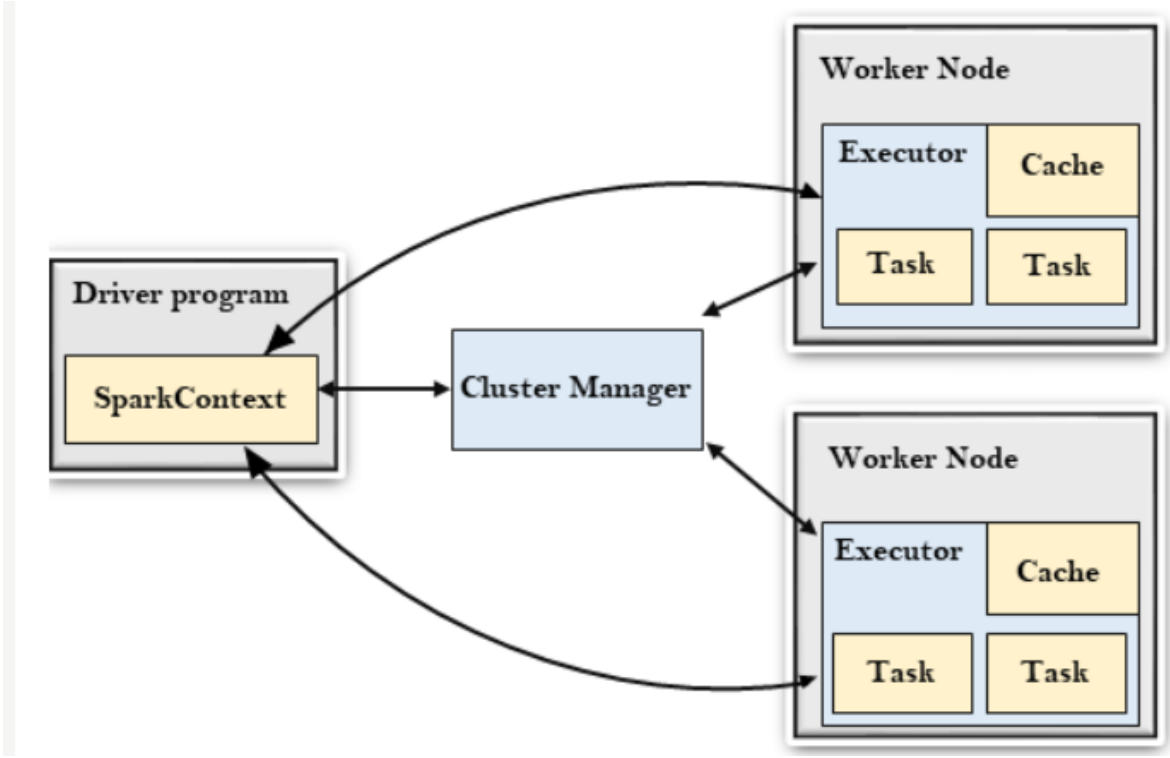
Le Framework Spark possède plusieurs fonctionnalités.

• Le Spark Core: c'est le système central de Spark. C'est une brique dédiée au traitement distribué des données (comme Hadoop MapReduce).

• Les modules avancés: Ils sont développés au-dessus de Spark Core et permettent de faire des traitements complexes (Streaming, machine learning, SQL…)



Mini Architecture



Mini Architecture

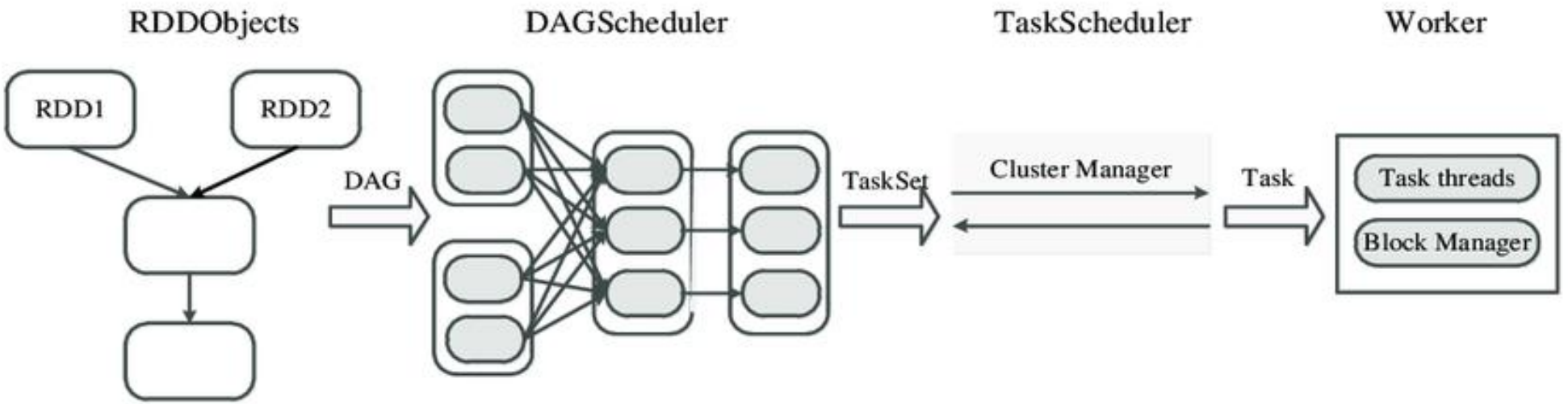
1.Driver (Spark Context) :
Le Driver est responsable de l'exécution du programme principal Spark. Il initialise le **Spark Context**, qui est l'interface pour communiquer avec le cluster. Le Driver divise le code en tâches, planifie leur exécution, et collecte les résultats finaux. Il assure également la coordination avec le Cluster Manager pour l'allocation des ressources.

2.Cluster Manager :
Le Cluster Manager est le composant qui gère les ressources du cluster. Il peut être de différents types, comme **YARN**, **Mesos**, ou **Standalone Spark Cluster Manager**. Le Cluster Manager attribue les ressources (mémoire et CPU) aux tâches en fonction des besoins du Driver.

3.Worker Node :
Les Worker Nodes sont les machines qui fournissent les ressources de calcul au cluster. Chaque Worker peut héberger un ou plusieurs **Executors** qui effectuent les tâches assignées. Les Worker Nodes reçoivent les instructions du Driver et les exécutent sur leurs ressources disponibles.

4.Executor :
Les Executors sont des processus qui tournent sur les Worker Nodes et exécutent les tâches. Ils effectuent le traitement des données et stockent les résultats intermédiaires en mémoire ou sur disque. Chaque tâche Spark est exécutée par un Executor, qui communique ensuite les résultats au Driver.

Exécution


RDDObjects
DAGScheduler
TaskScheduler
Worker
RDD1
RDD2
DAG
TaskSet
Cluster Manager
Task
Task threads
Block Manager

Opérartions

Lazy Evaluation : L'évaluation paresseuse est une stratégie d'évaluation qui maintient l'évaluation d'une expression jusqu'à ce que sa valeur soit nécessaire. Cela évite l'évaluation répétée ce qui permet une optimisation des étapes du traitement.

Les transformations : Ce sont des fonctions qui retournent une nouvelle SD(structure de données) . Rien n'est évalué lorsque l'on fait appel à une fonction de transformation, cette fonction prend juste une nouvelle et retourne un nouveau SD. Les fonctions de transformation sont par exemple : map, filter, flatMap, groupByKey, reduceByKey, aggregateByKey, Exp: df.filter(_.age > 21),

Nous avons deux type de transformation Narrow opération qui nécessitent un shuffle après un stage et Wide operation qui n' ont pas besoin

Les actions : les actions évaluent et retournent une nouvelle valeur. Au moment où une fonction d'action est appelée sur une SD, toutes les requêtes de traitement des données sont calculées et le résultat est retourné. Les actions sont par exemple : reduce, collect, count, first, take, countByKey et foreach

Autres: **persist**, **cache, broadcast**, …….

Les Types de données

- RDD (Resilient Distributed Dataset) -depuis la version 1.0.

Un RDD est une collection distribuée immutable de données calculée à partir d'une source et conservée en mémoire vive (tant que la capacité le permet). Les données sont organisées en objets par ligne .

- DataFrames - Spark introduit des DataFrames dans la version 1.3

Un DataFrame est une collection distribuée immutable de données. Contrairement à un RDD, les données sont organisées en colonnes nommées, comme une table dans une base de données relationnelle.

- DataSet - Spark a introduit le jeu de données dans 1.6. DataSet =Dataframe+RDD
- Discretized Stream (DStream): est une séquence continue de RDD (du même type) , une fonction qui est utilisée pour
- générer un RDD après chaque intervalle de temps

!!!!Dans ce cours on se limitera que sur les RDD et DF

Application1

Objectif du TP :
L'objectif de ce TP est d'utiliser les RDD pour effectuer des opérations de base sur un ensemble de données contenant des informations sur des étudiants.

Prérequis :
Vous devez avoir IntelliJ

Énoncé du TP :
Vous disposez d'un fichier texte contenant des données sur les étudiants. Chaque ligne du fichier contient les informations
https://github.com/idiattara/sda_2024/blob/main/student.txt

Votre tâche consiste à utiliser Spark RDD pour répondre aux questions suivantes :

1. Lire les données depuis un fichier texte en tant qu'RDD. (Nom du fichier : "students.csv")
2. Combien d'étudiants sont dans le RDD ?
3. Affichez les 5 premiers étudiants du RDD.
4. Calculez l'âge moyen des étudiants.
5. Trouvez le nombre d'étudiants par ville.
6. Trouvez le plus jeune étudiant.

Correction: https://github.com/idiattara/sda_2024/blob/main/correction_app1.scala

Application Dataset

Vous disposez d'un fichier texte contenant des données de ventes sous la forme suivante :

https://github.com/idiattara/Spark_DIATTARA/blob/main/sales.txt

Votre tâche consiste à utiliser les Datasets Spark pour répondre aux questions suivantes :

1. Lire les données depuis le fichier texte en tant que Dataset. (Nom du fichier : "sales.csv")
2. Affichez les 5 premières lignes du Dataset.
3. Calculez le montant total des ventes pour chaque produit.
4. Trouvez le produit le plus vendu.
5. Calculez la somme totale des ventes pour chaque vendeur.

Correction https://github.com/idiattara/Spark_DIATTARA/blob/main/correction_App2.scala

DataFrame

Sera réaliser sous Databricks
Correction :

https://github.com/idiattara/Spark_DIATTARA/blob/main/correction_tp_dataframe.html