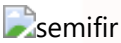
Spark : Architecture

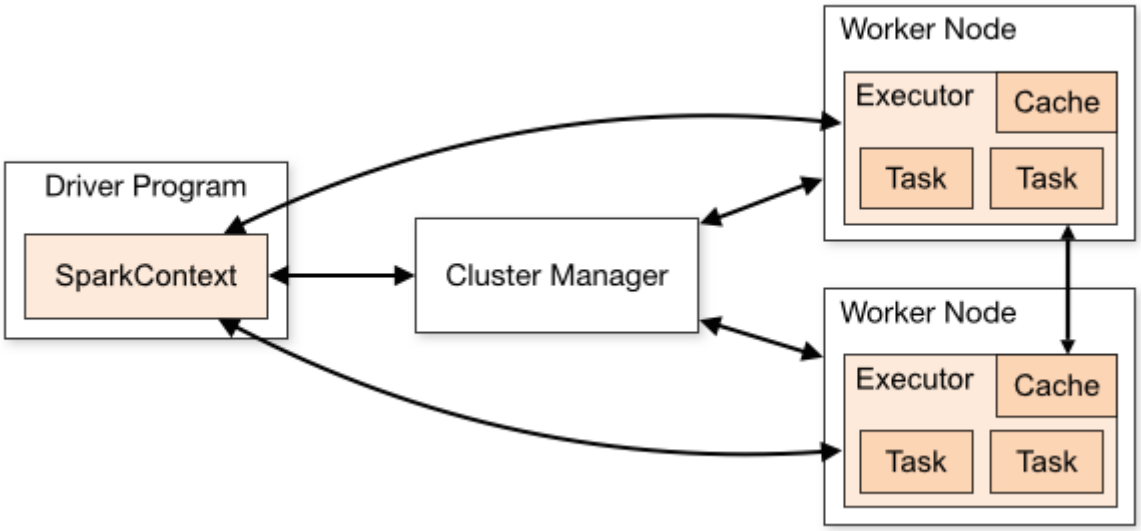


Spark : Architecture

Spark est un systeme de traitement de traitement distribué.

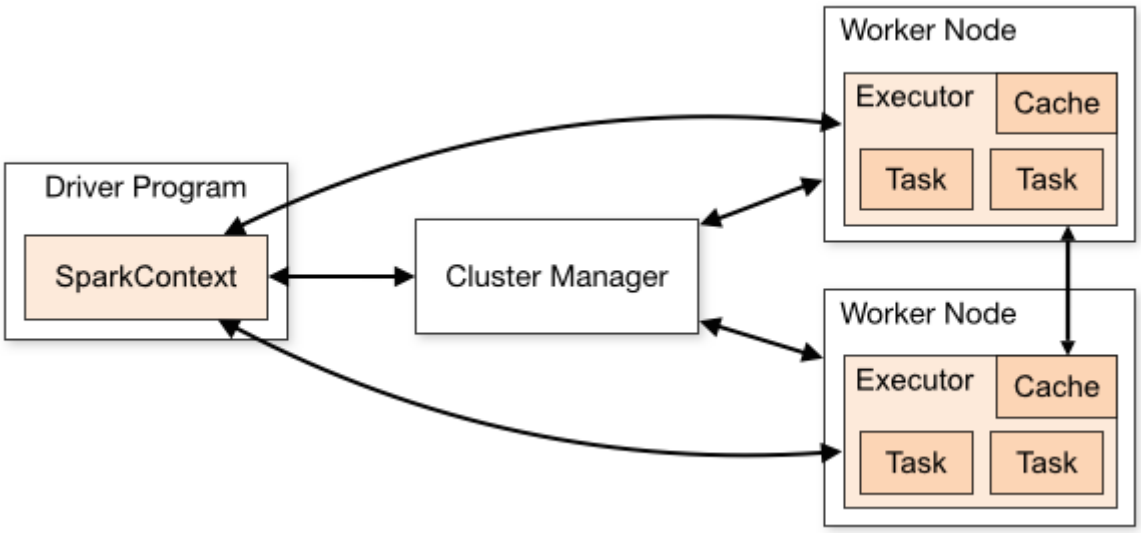
Il necessite de nombreux composants pour fonctionner.

Ils sont réunies dans un cluster, c'est à dire un ensemble de machines qui communiquent entre elles.



Le travail de spark se base sur une architecture de type **maitre-esclave** :

- Le maitre est le point d'entrée de l'application spark
- Les esclaves sont les machines de traitementement



Dans cette architecture, on trouve :

Le driver, les taches à accomplir, les workers, le cluster manager, le contexte, ect...

Le Driver

> C'est le point d'entrer de l'application spark.

Le noeud driver est le coordinateur de travail à effectuer.

C'est lui qui cree et detient le contexte spark responsable de la soumission des taches aux workers.

Note: sorte de chef qui planifie et suit l'execution des taches.

Les Workers

> Ce sont les noeuds machines qui effectuent le travail.

Un worker possende une memoire dediée au stockage des données et des partitions de travail.

C'est lui qui effectue le travail d'une tache à accomplir.

Le Cluster Manager

> C'est le gestionnaire de ressources du cluster.

Le cluster manager est responsable de gerer l'etat des workers, la repartitions et la planifications des taches.

Le Cluster Manager

Spark eut utiliser different cluster manager :

- Spark Standalone (manager locale)

- Apache Mesos (manager distribué)
- Hadoop YARN (manager distribué)
- Kubernetes (manager distribué)

Le contexte

> C'est l'interface entre le driver et les workers.

Il permet de **parrametrer le cluster** et de soumettre les taches aux workers avec l'aide du cluster manager.

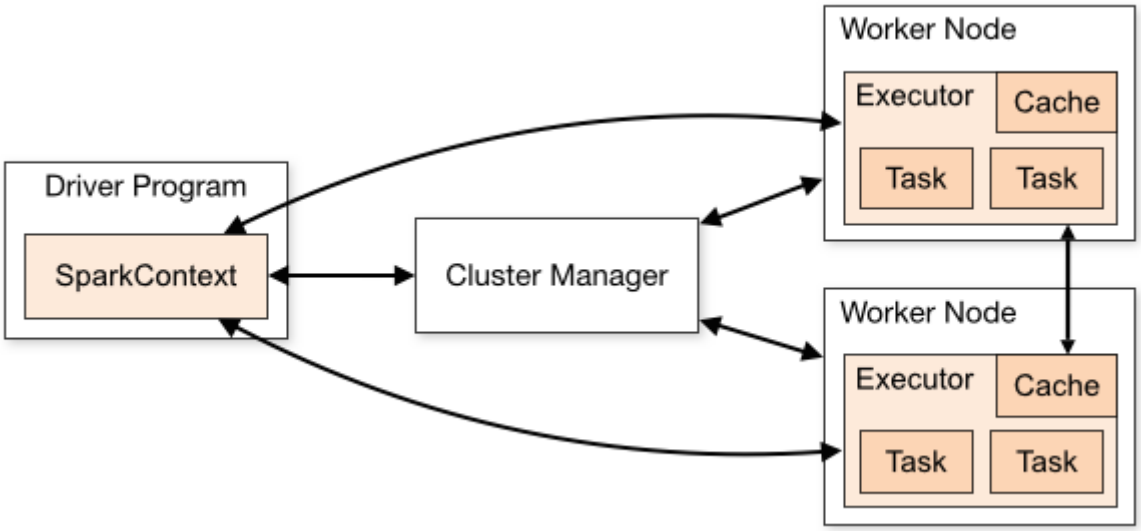
Le contexte le point d'entrée qui recupere les données à traiter et le travail à effectuer.

Une tache

> C'est une **unité de traitement** de données.

Une tache est creer par le driver à partir des données sources et du programme à executer.

L'execusion d'un programme est donc une suite de taches à effectuer sur une suite de données.



- Le **driver** reçoit les données et le traitement
- Le **contexte** definit le cluster et les taches à effectuer
- Les taches sont réparties sur les workers sous la supervision du **cluster manager**
- Les **workers** effectuent les taches et retournent les resultats au driver.

Fonctionement