Le Principe des Services Globaux et Répartis dans Docker Swarm

Docker Swarm permet de gérer des services conteneurisés à grande échelle, de manière orchestrée. Deux types principaux de services peuvent être utilisés pour déployer des conteneurs sur un cluster Docker Swarm : **les services globaux** et **les services répartis**. Chacun a des caractéristiques spécifiques en fonction des besoins de scalabilité, de disponibilité et de distribution des conteneurs dans le cluster.

---

1. Différences entre Services Globaux et Répartis

a. Services Globaux

Les **services globaux** sont des services qui sont déployés sur **chaque nœud** du cluster Docker Swarm. Cela signifie que Docker Swarm va garantir qu'il y a une instance (réplica) du conteneur pour chaque nœud disponible dans le cluster.

Caractéristiques des services globaux :

- Chaque nœud reçoit une instance du service. Par exemple, si vous avez 5 nœuds dans votre cluster, Docker créera 5 instances du service, une sur chaque nœud.
- Les services globaux sont utilisés lorsque vous avez besoin de déployer un service sur tous les nœuds sans exception, ce qui est typiquement le cas pour des tâches comme le monitoring ou la gestion des logs, où chaque nœud doit exécuter un conteneur.
- Il n'y a pas de notion de réplication manuelle ici. Chaque nœud exécutera une instance du service de manière automatique.

Cas d'utilisation des services globaux :

- **Services de gestion** : Par exemple, des agents de monitoring ou de logging, où chaque nœud doit avoir son propre conteneur qui collecte des données localement.
- **Services stateless** : Lorsque vous avez des services qui ne nécessitent pas de réplication ou d'état partagé entre les conteneurs.

b. Services Répartis

Les **services répartis** (ou **replicated services**) sont des services où Docker Swarm **répartit les instances** du service sur les nœuds du cluster selon un nombre spécifié de réplicas. Vous définissez combien d'instances du service doivent être déployées, et Docker Swarm s'assure de déployer ces instances sur les nœuds disponibles dans le cluster.

Caractéristiques des services répartis :

- Vous spécifiez un nombre de réplicas pour le service, et Docker répartit automatiquement ces réplicas sur les différents nœuds.
- Si un nœud échoue, Docker Swarm redémarrera les réplicas manquants sur d'autres nœuds, garantissant ainsi une haute disponibilité du service.

- Contrairement aux services globaux, les services répartis n'ont pas nécessairement une instance sur chaque nœud. Le nombre de réplicas est déterminé par la configuration, et Swarm s'assure que ces réplicas sont déployés de manière optimale.

Cas d'utilisation des services répartis :

- **Applications scalables** : Par exemple, une application web où plusieurs réplicas sont nécessaires pour gérer la charge.
- **Haute disponibilité** : Lorsque vous souhaitez garantir qu'un nombre précis d'instances du service soit disponible à tout moment, même en cas de défaillance d'un nœud.

---

2. Comment Utiliser Ces Services dans un Cluster Swarm

a. Créer un Service Global

Pour créer un service global, vous utilisez la commande `docker service create` avec l'option `--mode global`. Cela déploie le service sur tous les nœuds du cluster.

Exemple : Déployer un service global qui exécute un conteneur Nginx sur chaque nœud du cluster.

```
docker service create --name nginx-global --mode global nginx
```

- `--mode global` : Cette option indique que le service doit être déployé sur chaque nœud du cluster.
- `nginx` : Le nom de l'image Docker à utiliser.

Dans cet exemple, Docker Swarm va s'assurer qu'il y a une instance du service Nginx sur chaque nœud du cluster.

b. Créer un Service Réparti (avec des réplicas)

Pour créer un service réparti, vous utilisez la commande `docker service create` avec l'option `--replicas`, où vous spécifiez le nombre de réplicas que vous souhaitez déployer.

Exemple : Déployer un service réparti avec 3 réplicas de Nginx :

```
docker service create --name nginx-replicated --replicas 3 nginx
```

- `--replicas 3` : Cette option définit que 3 réplicas de Nginx doivent être créés et répartis sur les nœuds disponibles du cluster.

Si vous ajoutez ou supprimez des nœuds dans votre cluster, Docker Swarm ajuste automatiquement le nombre d'instances de Nginx pour maintenir 3 réplicas.

c. Mise à l'Échelle d'un Service Réparti

Vous pouvez également modifier dynamiquement le nombre de réplicas d'un service réparti en utilisant la commande `docker service scale`.

Exemple : Pour augmenter le nombre de réplicas de 3 à 5 pour un service `nginx-replicated` :

```
docker service scale nginx-replicated=5
```

Docker Swarm gère automatiquement la mise à l'échelle du service en créant ou supprimant des réplicas selon les besoins.

d. Inspection et Gestion des Services

- **Vérifier les services dans le Swarm** :
  Pour voir l'état des services dans le cluster, vous pouvez utiliser :

  ```
  docker service ls
  ```

- **Vérifier les détails d'un service** :
  Pour voir les détails d'un service spécifique, y compris le nombre de réplicas et leur état, vous pouvez utiliser :

  ```
  docker service ps <service_name>
  ```

Résumé des Différences entre Services Globaux et Répartis

Type de Service	Déploiement	Cas d'Utilisation
Service Global	Une instance par nœud du cluster	Tâches comme le monitoring, le logging, ou tout autre service qui doit être présent sur chaque nœud
Service Réparti	N instances réparties sur les nœuds	Applications nécessitant des réplicas pour la scalabilité et la haute disponibilité (ex. API, sites web)

Conclusion

- **Services Globaux** : Parfait pour déployer un service sur chaque nœud du cluster, sans avoir besoin de spécifier combien d'instances doivent être créées. Cela garantit que le service est disponible sur tous les nœuds.
- **Services Répartis** : Utilisé pour déployer un service avec un nombre défini de réplicas qui seront répartis sur les nœuds du cluster. Ces services sont idéaux pour les applications nécessitant la

scalabilité et la haute disponibilité.

Docker Swarm permet ainsi une gestion flexible des services, en offrant des solutions adaptées à des besoins différents, que ce soit pour des services qui doivent être présents partout ou pour des services nécessitant une mise à l'échelle et une gestion optimisée des ressources.