TP n°3, Partitionnement sous MongoDB

Partie 1 : Mise en place du partitionnement :

Mise en place du sharding :

Étant donné que nous avons déjà installé MongoDB, nous allons simplement démarrer l'instance dans Docker puis nous connecter via le terminal. Ici, nous avons besoin de 6 terminaux : un pour la configuration, un pour le routeur, un pour le client, un pour l'initialisation du replica set et enfin 2 pour les serveurs. Nous allons les nommer en conséquence afin de pouvoir nous y retrouver plus facilement.

Une fois connecté, nous allons dans un premier temps créer des répertoire de stockage. Un pour le répertoire et deux autres pour les shards.

On va ensuite démarrer le serveur de configuration :
```
configsrv#mongod --configsvr --replSet replicaconfig --dbpath configsvrdb --port 27019
```

On va ensuite l'initialiser.

On lance le routeur :
```
routeur# mongos --configdb replicaconfig/localhost:27019
```

On démarre les deux shards :
```
serv1#mongod --replSet replicashard1 --dbpath serv1/ --shardsvr --port 20004
serv2#mongod --replSet replicashard2 --dbpath serv2/ --shardsvr --port 20005
```

Afin d'ajouter ces deux shards, nous allons rentrer les commandes suivantes sur le routeur :
```
routeur#sh.addShard("replicashard1/localhost:20004")
routeur#sh.addShard("replicashard2/localhost:20005")
```

Puis on active le sharding :
```
routeur#sh.enableSharding("mabasefilms")
```

Enfin on active le sharding sur la collection :
```
routeur#sh.shardCollection("mabasefilms.films", { "titre": 1 })
```

Nous avons mis en place le partitionnement pour notre collection que l'on pourra remplir par la suite à l'aide d'un programme python.

Partie 2 : Questions concernant le sharding :

1. Qu'est-ce que le sharding dans MongoDB et pourquoi est-il utilisé ?

Le sharding dans MongoDB consiste à répartir les données d'une collection sur plusieurs serveurs appelés shards.
Il est utilisé pour gérer de grands volumes de données et améliorer les performances et la scalabilité.

2. Quelle est la différence entre le sharding et la réplication dans MongoDB ?

Le sharding sert à répartir les données sur plusieurs serveurs pour gérer de gros volumes et améliorer les performances.
La réplication sert à dupliquer les données sur plusieurs serveurs pour assurer la disponibilité et la tolérance aux pannes.

3. Quels sont les composants d'une architecture shardée (mongos, config servers, shards) ?

Les shards contiennent les données, les config servers stockent les informations de configuration du cluster, et mongos est le routeur qui redirige les requêtes des clients vers les bons shards.

4. Quelles sont les responsabilités des config servers (CSRS) dans un cluster shardé ?

Les config servers (CSRS) stockent les métadonnées du cluster shardé, comme la répartition des données et la configuration des shards.
Ils permettent à MongoDB de savoir où se trouvent les données et comment les requêtes doivent être routées.

5. Quel est le rôle du mongos router ?

Le mongos est le routeur du cluster shardé : il reçoit les requêtes des clients et les redirige vers les shards concernés.
Il rassemble ensuite les résultats et les renvoie au client.

6. Comment MongoDB décide-t-il sur quel shard stocker un document ?

MongoDB utilise la clé de shard du document pour décider sur quel shard le stocker.
La valeur de cette clé détermine le shard en fonction des plages de données ou du hachage.

7. Qu'est-ce qu'une clé de sharding et pourquoi est-elle essentielle ?

Une clé de sharding est un champ (ou ensemble de champs) utilisé pour répartir les documents entre les shards.
Elle est essentielle car elle détermine la distribution des données et les performances des requêtes.

8. Quels sont les critères de choix d'une bonne clé de sharding ?

Une bonne clé de sharding doit avoir une forte cardinalité et permettre une répartition équilibrée des données.
Elle doit aussi être souvent utilisée dans les requêtes pour éviter de consulter tous les shards.

9. Qu'est-ce qu'un chunk dans MongoDB ?

Un chunk est une portion de données d'une collection shardée.

10. Comment fonctionne le splitting des chunks ?

Le splitting se produit lorsqu'un chunk devient trop volumineux.
MongoDB le divise automatiquement en plusieurs chunks plus petits en fonction de la clé de sharding.

11. Que fait le balancer dans un cluster shardé ?

Le balancer répartit les chunks de manière équilibrée entre les shards.
Il déplace automatiquement les chunks pour éviter qu'un shard soit surchargé.

12. Quand et comment le balancer déplace-t-il des chunks ?

Le balancer agit lorsqu'il détecte un déséquilibre entre les shards.
Il déplace alors automatiquement des chunks d'un shard trop chargé vers un shard moins chargé.

13. Qu'est-ce qu'un hot shard et comment l'éviter ?

Un hot shard est un shard qui reçoit beaucoup plus de requêtes ou d'écritures que les autres, ce qui crée un déséquilibre.
Cela arrive souvent à cause d'une mauvaise clé de sharding (par exemple des valeurs croissantes).
On peut l'éviter en choisissant une clé qui répartit uniformément les données et les accès, comme une clé hachée.

14. Quels problèmes une clé de sharding monotone peut-elle engendrer ?

Une clé de sharding monotone (comme une date ou un identifiant croissant) peut provoquer un déséquilibre des écritures, car les nouveaux documents vont toujours vers le même shard.
Cela peut créer un hot shard et dégrader les performances du cluster.

15. Comment activer le sharding sur une base de données et sur une collection ?

On active le sharding sur une base de données avec la commande `sh.enableSharding("nomBase").`
Ensuite, on shard une collection avec `sh.shardCollection("nomBase.nomCollection", { cle: 1 })`, en précisant la clé de sharding.

16. Comment ajouter un nouveau shard à un cluster MongoDB ?

On ajoute un nouveau shard avec la commande `sh.addShard()` depuis un mongos, en indiquant l'adresse du serveur ou du replica set à ajouter au cluster.

17. Comment vérifier l'état du cluster shardé (commandes usuelles) ?

On peut vérifier l'état du cluster avec la commande `sh.status()`, qui affiche les shards, les bases et la répartition des chunks.

18. Dans quels cas faut-il envisager d'utiliser un hashed sharding key ?

Un hashed sharding key est utile quand les valeurs de la clé sont monotones ou très concentrées, afin d'assurer une répartition uniforme des données et des écritures entre les shards.

19. Dans quels cas faut-il privilégier un ranged sharding key ?

Un ranged sharding key est à privilégier quand on fait souvent des requêtes par plages de valeurs (par exemple des dates), car il permet de cibler seulement les shards concernés.

20. Qu'est-ce que le zone sharding et quel est son intérêt ?

Le zone sharding permet d'associer certaines plages de la clé de sharding à des shards spécifiques.
Il est utile pour contrôler la localisation des données, par exemple pour des contraintes géographiques ou de performance.

21. Comment MongoDB gère-t-il les requêtes multi-shards ?

MongoDB envoie la requête via mongos à tous les shards concernés.
Les résultats sont ensuite regroupés et renvoyés au clien**t.**

22. Comment optimiser les performances de requêtes dans un environnement shardé ?

On optimise les performances en choisissant une bonne clé de sharding utilisée dans les requêtes et en indexant correctement les champs.
Cela permet à MongoDB de cibler un minimum de shards au lieu d'interroger tout le cluster.

23. Que se passe-t-il lorsqu'un shard devient indisponible ?

Si un shard devient indisponible, les données qu'il contient ne sont plus accessibles, mais si le shard est un replica set, MongoDB peut continuer à fonctionner grâce à un nœud secondaire.

24. Comment migrer une collection existante vers un schéma shardé ?

On active d’abord le sharding sur la base, puis on choisit une clé de sharding et on utilise sh.shardCollection() pour transformer la collection existante en collection shardée.

25. Quels outils ou métriques utiliser pour diagnostiquer les problèmes de sharding ?

On peut utiliser sh.status(), les métriques de MongoDB Atlas ou de mongostat et mongotop pour analyser la répartition des chunks, la charge des shards et les performances.