Seyfeddine Jouini 3ING

Q&A - Réplication avec MongoDB

Partie 1 - Compréhension de base

1. Qu'est-ce qu'un Replica Set dans MongoDB ?
Un Replica Set est un groupe de serveurs MongoDB maintenant automatiquement des copies synchronisées des mêmes données.

2. Quel est le rôle du Primary dans un Replica Set ?
Le Primary accepte toutes les écritures et sert de source principale de réplication pour les autres nœuds.

3. Quel est le rôle essentiel des Secondaries ?
Les Secondaries répliquent les données du Primary pour assurer la redondance et éventuellement servir des lectures.

4. Pourquoi MongoDB n'autorise-t-il pas les écritures sur un Secondary ?
MongoDB interdit les écritures sur un Secondary pour éviter les conflits de données et garantir une cohérence stable.

5. Qu'est-ce que la cohérence forte dans le contexte MongoDB ?
La cohérence forte signifie que toute lecture reflète immédiatement la dernière écriture validée par le Primary.

6. Quelle est la différence entre readPreference : "primary" et "secondary" ?
"primary" lit toujours les données à jour tandis que "secondary" peut lire des données en retard.

7. Dans quel cas pourrait-on souhaiter lire sur un Secondary malgré les risques ?
On lit sur un Secondary pour répartir la charge ou effectuer des analyses tolérant un léger décalage

Partie 2 - Commandes & configuration

8. Quelle commande permet d'initialiser un Replica Set ?
La commande d'initiation d'un Replica Set est `rs.initiate()`.

9. Comment ajouter un nœud à un Replica Set après son initialisation ?
On ajoute un nœud avec la commande `rs.add("hostname:port")`.

10. Quelle commande permet d'afficher l'état actuel du Replica Set ?
L'état du Replica Set s'affiche avec `rs.status()`.

11. Comment identifier le rôle actuel (Primary / Secondary / Arbitre) d'un nœud ?
On identifie le rôle via `rs.status()` ou `db.hello()`.

12. Quelle commande permet de forcer le basculement du Primary ?
Le basculement se force avec `rs.stepDown()`.

13. Comment peut-on désigner un nœud comme Arbitre ? Pourquoi le faire ?
On désigne un arbitre avec `rs.addArb("hostname:port")` pour ajouter un vote sans stocker de données.

14. Donnez la commande pour configurer un nœud secondaire avec un délai de réplication (slaveDelay).
On configure un délai via `rs.reconfig()` en définissant `"slaveDelay" : <secondes>`.

Partie 3 - Résilience et tolérance aux pannes

15. Que se passe-t-il si le Primary tombe en panne et qu'il n'y a pas de majorité ?
Sans majorité, aucun Primary ne peut être élu et le cluster devient en lecture seule.

16. Comment MongoDB choisit-il un nouveau Primary ? Quels critères utilise-t-il ?
MongoDB élit un Primary selon la priorité, le vote de la majorité et la fraîcheur des données répliquées.

17. Qu'est-ce qu'une élection dans MongoDB ?
Une élection est le processus par lequel les nœuds votent pour choisir un nouveau Primary.

18. Que signifie auto-dégradation du Replica Set ? Dans quel cas cela survient-il ?
L'auto-dégradation signifie qu'un Primary se transforme en Secondary lorsqu'il perd la majorité.

19. Pourquoi est-il conseillé d'avoir un nombre impair de nœuds dans un Replica Set ?
Un nombre impair facilite l'obtention d'une majorité et limite les blocages.

20. Quelles conséquences a une partition réseau sur le fonctionnement du cluster ?
Une partition réseau peut empêcher la majorité et rendre certaines parties du cluster en lecture seule.

Partie 4 - Scénarios pratiques

21. Vous avez 3 nœuds : 27017 (Primary), 27018 (Secondary), et 27019 (Arbitre). Que se passe-t-il si le Primary devient injoignable ?
Si le Primary tombe, le Secondary et l'Arbitre forment une majorité et élisent un nouveau Primary.

22. Vous avez configuré un Secondary avec un slaveDelay de 120 secondes : quelle est son utilité ?
Un Secondary avec un délai de 120 secondes permet de disposer d'un historique utile pour restaurer des erreurs récentes.

23. Un client exige une lecture toujours à jour, même en cas de bascule : quelles options recommander ?
Il faut utiliser `readConcern: "majority"` et `writeConcern: { w: "majority" }`.

24. Dans une application critique, vous voulez garantir que l'écriture est confirmée par au moins deux nœuds : quel writeConcern utiliser ?
Il faut utiliser `writeConcern: { w: 2 }`.

25. Un étudiant a lu depuis un Secondary et récupéré une donnée obsolète : pourquoi et comment éviter cela ?
Le Secondary peut être en retard, et on évite cela en lisant sur le Primary ou avec `readConcern: "majority"`.

26. Montrez la commande pour vérifier quel nœud est actuellement Primary.
On utilise `rs.status()` ou `db.hello()`.

27. Expliquez comment forcer une bascule manuelle du Primary sans interruption majeure.
On force la bascule en exécutant `rs.stepDown()` sur le Primary.

28. Décrivez la procédure pour ajouter un nouveau nœud secondaire dans un Replica Set en fonctionnement.
Il suffit de lancer MongoDB sur le nouveau serveur puis d'exécuter `rs.add("host:port")`.

29. Quelle commande permet de retirer un nœud défectueux d'un Replica Set ?
On le retire avec `rs.remove("host:port")`.

30. Comment configurer un nœud secondaire pour qu'il soit caché (non visible aux clients) ? Pourquoi ?
On le cache avec `"hidden": true` dans `rs.reconfig()` pour éviter qu'il soit utilisé par les applications.

31. Montrez comment modifier la priorité d'un nœud afin qu'il devienne le Primary préféré.
On ajuste sa priorité dans la configuration via `rs.reconfig()` en modifiant `"priority" : <valeur>`.

32. Expliquez comment vérifier le délai de réplication d'un Secondary par rapport au Primary.
On vérifie le délai avec `rs.printSlaveReplicationInfo()`.

33. Que fait la commande rs.freeze() et dans quel scénario est-elle utile ?
`rs.freeze()` empêche un nœud de devenir Primary pendant un temps donné pour stabiliser une élection.

34. Comment redémarrer un Replica Set sans perdre la configuration ?
Il suffit de redémarrer chaque service MongoDB, la configuration étant stockée dans les données.

35. Expliquez comment surveiller en temps réel la réplication via les logs MongoDB ou commandes shell.
On surveille la réplication via `rs.status()`, `rs.printReplicationInfo()`, et les logs mongod.

Questions complémentaires

37. Qu'est-ce qu'un Arbitre (Arbiter) et pourquoi ne stocke-t-il pas de données ?
Un Arbitre sert uniquement à voter pour les élections et ne stocke aucune donnée pour rester léger.

38. Comment vérifier la latence de réplication entre le Primary et les Secondaries ?

On vérifie la latence via `rs.printSlaveReplicationInfo()`.

39. Quelle commande MongoDB permet d'afficher le retard de réplication des membres secondaires ?

La commande est `rs.printSlaveReplicationInfo()`.

40. Quelle est la différence entre la réplication asynchrone et synchrone ? Quel type utilise MongoDB ?

La réplication synchrone attend tous les nœuds, l'asynchrone non, et MongoDB utilise une réplication asynchrone.

41. Peut-on modifier la configuration d'un Replica Set sans redémarrer les serveurs ?

Oui, on peut modifier la configuration avec `rs.reconfig()` sans redémarrer les nœuds.

42. Que se passe-t-il si un nœud Secondary est en retard de plusieurs minutes ?

Il peut devenir inutilisable pour certaines lectures et parfois nécessiter une resynchronisation complète.

43. Dans quelles conditions MongoDB gère-t-il les conflits de données lors de la réplication ?

MongoDB applique la règle du "last write wins" et remplace les données du Secondary par celles du Primary.

44. Est-il possible d'avoir plusieurs Primarys simultanément dans un Replica Set ? Pourquoi ?

Non, car le consensus RAFT exige une seule source d'écriture pour éviter les incohérences.

45. Pourquoi est-il déconseillé d'utiliser un Secondary pour des opérations d'écriture même en lecture préférée secondaire ?

Parce que les écritures sur les Secondaries ne sont pas permises et entraîneraient une perte de cohérence.

46. Quelles sont les conséquences d'un réseau instable sur un Replica Set ?

Un réseau instable provoque des réélections fréquentes, des pertes de Primary et des interruptions d'écriture.