

Nom : Loubna MANGOUCI

N°Étudiant : 124099931

Classe : INFO3

Réplication et tolérance aux pannes avec MongoDB

Partie 1 : Compréhension de base

1. Qu'est-ce qu'un Replica Set dans MongoDB ?

Un groupe de serveurs MongoDB maintenant une copie identique des données.

2. Quel est le rôle du Primary dans un Replica Set ?

Accepter toutes les écritures et servir les lectures par défaut.

3. Quel est le rôle essentiel des Secondaries ?

Répliquer les données du Primary et servir éventuellement les lectures.

4. Pourquoi MongoDB n'autorise-t-il pas les écritures sur un Secondary ?

Pour garantir la cohérence et éviter les conflits.

5. Qu'est-ce que la cohérence forte dans le contexte MongoDB ?

Garantit que toute lecture obtient la donnée la plus récente.

6. Quelle est la différence entre readPreference : "primary" et "secondary" ?

Primary : lecture toujours à jour.

Secondary : lecture sur un nœud répliqué (risque de retard).

7. Dans quel cas pourrait-on souhaiter lire sur un Secondary malgré les risques ?

Pour répartir la charge ou pour des besoins analytiques.

Partie 2 : Commandes & configuration

8. Quelle commande permet d'initialiser un Replica Set ?

`rs.initiate()`

9. Comment ajouter un nœud à un Replica Set après son initialisation ?

`rs.add("host:port")`

10. Quelle commande permet d'afficher l'état actuel du Replica Set ?

`rs.status()`

11. Comment identifier le rôle actuel (Primary / Secondary / Arbitre) d'un nœud ?

Avec `rs.status()` en lisant le champ `stateStr`.

12. Quelle commande permet de forcer le basculement du Primary ?

`rs.stepDown()`

13. Comment peut-on désigner un nœud comme Arbitre ? Pourquoi le faire ?

`rs.addArb("host:port")` → Pour participer à l'élection du primaire en cas de panne, sans stocker de données.

14. Donnez la commande pour configurer un nœud secondaire avec un délai de réplication (`slaveDelay`).

`cfg.members[i].slaveDelay = 120 ; rs.reconfig(cfg)`

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 ?
Chaque sous-réseau va définir son Primary -> Conflit

16. Comment MongoDB choisit-il un nouveau Primary ? Quels critères utilise-t-il ?
MongoDB élit un nouveau Primary en fonction de :

- la majorité des votes,
- la priorité configurée,
- la disponibilité et l'état des nœuds.

17. Qu'est-ce qu'une élection dans MongoDB ?

Un processus automatique permettant de choisir un nouveau Primary lorsqu'un Primary tombe ou devient injoignable.

18. Que signifie auto-dégradation du Replica Set ? Dans quel cas cela survient-il ?
C'est lorsqu'un Primary se rétrograde automatiquement en Secondary parce qu'il perd la majorité ou détecte qu'un autre Primary a été élu.

19. Pourquoi est-il conseillé d'avoir un nombre impair de nœuds dans un Replica Set ?
Pour faciliter l'obtention d'une majorité lors des votes, et éviter les égalités.

20. Quelles conséquences a une partition réseau sur le fonctionnement du cluster ?
Une partition sans majorité empêche l'élection d'un Primary, rendant le groupe inapte à écrire. → Seul le sous-réseau détenant la majorité peut élire un Primary : les autres restent en Secondary.

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 ?
Le Secondary devient Primary grâce au vote de l'arbitre.

22. Vous avez configuré un Secondary avec un slaveDelay de 120 secondes.
Quelle est son utilité ? Quels usages peut-on en faire dans la vraie vie ?
Avoir une copie retardée pour se protéger des suppressions accidentelles.

23. Un client exige une lecture toujours à jour, même en cas de bascule.
Quelles options de readConcern et writeConcern recommanderiez-vous ?
readConcern: "majority" ; writeConcern: { w: "majority" }

24. Dans une application critique, vous voulez garantir que l'écriture est confirmée par au moins deux nœuds. Quelle option de writeConcern devez-vous utiliser ?
writeConcern: { w: 2 }

25. Un étudiant a lu depuis un Secondary et récupéré une donnée obsolète. Expliquez pourquoi et comment éviter cela.
Réplication asynchrone donc retard → Solution : readPreference: "primary" ou majority.

26. Montrez la commande pour vérifier quel nœud est actuellement Primary dans votre Replica Set.

```
rs.status()
```

27. Expliquez comment forcer une bascule manuelle du Primary sans interruption majeure.

```
rs.stepDown()
```

28. Décrivez la procédure pour ajouter un nouveau nœud secondaire dans un Replica Set en fonctionnement.

```
rs.add("host:port")
```

29. Quelle commande permet de retirer un nœud défectueux d'un Replica Set ?

```
rs.remove("host:port")
```

30. Comment configurer un nœud secondaire pour qu'il soit caché (non visible aux clients) ? Pourquoi ferait-on cela ?

```
cfg.members[i].hidden = true ; rs.reconfig(cfg)
```

31. Montrez comment modifier la priorité d'un nœud afin qu'il devienne le Primary préféré.

```
cfg.members[i].priority = X
```

```
rs.reconfig(cfg)
```

32. Expliquez comment vérifier le délai de réplication d'un Secondary par rapport au Primary.

Avec `rs.status()` en lisant le champs `optime`.

33. Que fait la commande `rs.freeze()` et dans quel scénario est-elle utile ?

`rs.freeze()` empêche un nœud de participer aux élections pendant un temps donné.

Elle est utile pour éviter qu'un nœud instable ou en retard devienne Primary.

34. Comment redémarrer un Replica Set sans perdre la configuration ?

Il suffit de relancer chaque instance mongod avec le même paramètre `--replSet nomReplicaSet`

35. Expliquez comment surveiller en temps réel la réplication via les logs MongoDB ou commandes shell.

On peut surveiller la réplication en utilisant :

- les logs du serveur (`mongod.log`) qui montrent les opérations répliquées.
- `rs.status()` pour voir l'état des nœuds.
- `rs.printSecondaryReplicationInfo()` pour visualiser les retards de réplication.

Questions complémentaires

37. Qu'est-ce qu'un Arbitre (Arbiter) et pourquoi ne stocke-t-il pas de données ?

Un arbitre est un nœud qui participe uniquement aux votes pour élire un Primary. Il ne stocke pas de données.

38. Comment vérifier la latence de réplication entre le Primary et les Secondaries ?
Avec `rs.status()` en lisant le champs `optime`.

39. Quelle commande MongoDB permet d'afficher le retard de réplication des membres secondaires ?
`rs.printSecondaryReplicationInfo()`

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

- Synchrone : le Primary attend la confirmation des Secondaries.
- Asynchrone : le Primary écrit puis réplique ensuite.

→ MongoDB utilise la réplication asynchrone.

41. Peut-on modifier la configuration d'un Replica Set sans redémarrer les serveurs ?
Oui avec `rs.reconfig(cfg)`

42. Que se passe-t-il si un nœud Secondary est en retard de plusieurs minutes ?
Il peut entrer en état `RECOVERING` et effectuer une resynchronisation complète.

43. Comment MongoDB gère-t-il les conflits de données lors de la réplication ?
Le Primary est la source d'autorité → les Secondaries appliquent toujours son journal d'opérations (`oplog`).

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

Non. Le système de quorum empêche l'élection de plusieurs Primarys en même temps. Théoriquement, seul le sous-ensemble qui obtient la majorité des votes peut élire un Primary, ce qui empêche l'existence simultanée de deux maîtres et évite le phénomène de "split-brain".

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

Parce que si on l'autorisait, avec la réplication asynchrone ça se bloquerait à cause de l'attente de confirmation.

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

- pertes temporaires du Primary,
- élections fréquentes,
- retards de réplication,
- cluster parfois en mode dégradé.