# Bases de données NoSQL

Réplication et tolérance aux pannes avec MongoDB

Compte-rendu de TP

7 décembre 2025

## 1 Partie 1 — Compréhension de base

**1. Qu'est-ce qu'un Replica Set dans MongoDB ?** ***Réponse :*** *C'est un groupe d'instances* `mongod` *(serveurs) qui maintiennent le même jeu de données. Il assure la redondance (plusieurs copies) et la haute disponibilité (tolérance aux pannes).*

**2. Quel est le rôle du Primary dans un Replica Set ?** ***Réponse :*** *Le Primary est le seul nœud autorisé à accepter les opérations d'écriture. Par défaut, il traite aussi toutes les opérations de lecture pour garantir la cohérence. Il coordonne la réplication via son journal d'opérations (Oplog).*

**3. Quel est le rôle essentiel des Secondaries ?** ***Réponse :*** *Ils répliquent les données du Primary de manière asynchrone pour assurer la redondance. En cas de panne du Primary, ils peuvent participer à une élection pour devenir le nouveau Primary. Ils peuvent aussi servir aux lectures (si configuré).*

**4. Pourquoi MongoDB n'autorise-t-il pas les écritures sur un Secondary ?** ***Réponse :*** *Pour garantir une cohérence forte et éviter les conflits de données. Il ne doit y avoir qu'une seule source de vérité (le Primary).*

**5. Qu'est-ce que la cohérence forte dans le contexte MongoDB ?** ***Réponse :*** *Cela signifie que toute lecture renvoie la version la plus récente d'une donnée. C'est garanti par défaut en lisant et en écrivant uniquement sur le Primary.*

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

- `"primary"` *(défaut) : On lit uniquement sur le maître (donnée toujours à jour).*
- `"secondary"` *: On lit sur les esclaves. Cela permet de répartir la charge mais risque de renvoyer une donnée obsolète (réplication asynchrone).*

**7. Dans quel cas pourrait-on souhaiter lire sur un Secondary malgré les risques ?** ***Réponse :*** *Pour des opérations d'analyse lourdes (Analytics), du reporting, ou des tâches de fond qui ne nécessitent pas la donnée à la milliseconde près, afin de ne pas ralentir le Primary qui gère les transactions utilisateurs.*

## 2 Partie 2 — Commandes & configuration

**8. Quelle commande permet d'initialiser un Replica Set ?** ***Réponse :*** *`rs.initiate()`*

**9. Comment ajouter un nœud à un Replica Set après son initialisation ?** ***Réponse :*** *`rs.add("hostname:port")`*

**10. Quelle commande permet d'afficher l'état actuel du Replica Set ?** ***Réponse :*** *`rs.status()`*

**11. Comment identifier le rôle actuel (Primary / Secondary / Arbitre) d'un nœud ?** ***Réponse :*** *Dans la sortie de `rs.status()`, regarder le champ `stateStr` pour chaque membre, ou utiliser la commande `db.isMaster()` (alias `rs.hello()`).*

**12. Quelle commande permet de forcer le basculement du Primary ?** ***Réponse :*** *`rs.stepDown()` (Cette commande force le Primary actuel à devenir Secondary, déclenchant une élection).*

**13. Comment peut-on désigner un nœud comme Arbitre ? Pourquoi le faire ?** ***Réponse :*** *Commande : `rs.addArb("hostname:port")`. On le fait pour obtenir un nombre impair de votants (majorité stricte) à moindre coût, car l'arbitre ne stocke pas de données.*

**14. Donnez la commande pour configurer un nœud secondaire avec un délai de réplication (*secondaryDelaySeconds*).** ***Réponse :*** *Il faut reconfigurer le membre avec l'option `secondaryDelaySeconds` (anciennement `slaveDelay`) et `priority: 0`. (Note : Ce n'est pas une simple commande unique, mais une reconfiguration via `rs.reconfig`).*

## 3 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 ?** ***Réponse :*** *Le cluster passe en mode lecture seule (Read-Only). Aucun nœud ne peut être élu Primary, donc les écritures sont impossibles jusqu'à ce que la majorité soit rétablie.*

**16. Comment MongoDB choisit-il un nouveau Primary ? Quels critères utilise-t-il ?** ***Réponse :***

1. *La connectivité (le nœud est joignable).*
2. *La Priorité (définie dans la config).*
3. *La fraîcheur des données (celui qui a l'Oplog le plus récent).*

**17. Qu'est-ce qu'une élection dans MongoDB ?** ***Réponse :*** *C'est un algorithme de consensus distribué (basé sur Raft) où les nœuds éligibles votent pour désigner un nouveau Primary suite à la perte de contact avec l'ancien.*

**18. Que signifie auto-dégradation du Replica Set ? Dans quel cas cela survient-il ?** ***Réponse :*** *Cela signifie que le système continue de fonctionner mais en mode limité (par exemple, passage des nœuds en Secondary uniquement) si le quorum (majorité) est perdu suite à une partition réseau ou des pannes multiples.*

**19. Pourquoi est-il conseillé d'avoir un nombre impair de nœuds dans un Replica Set ?** ***Réponse :*** *Pour éviter les situations d'égalité lors des votes (split-vote) et s'assurer qu'une majorité stricte peut toujours être atteinte mathématiquement.*

**20. Quelles conséquences a une partition réseau sur le fonctionnement du cluster ?** ***Réponse :*** *Risque de "Split-Brain". Le groupe minoritaire cesse d'accepter les écritures (devient Secondary). Seul le groupe majoritaire élit un Primary et continue le service.*

## 4 Partie 4 — Scénarios pratiques

**21. Vous avez 3 nœuds : A (Primary), B, et C. Que se passe-t-il si le Primary devient injoignable ?** ***Réponse :*** *B et C détectent l'absence de "heartbeats". Ils lancent une élection. Si B et C peuvent communiquer, l'un d'eux sera élu nouveau Primary (car ils forment la majorité : 2 sur 3).*

**22. Vous avez configuré un Secondary avec un *secondaryDelaySeconds* de 120 secondes. Quelle est son utilité ? Quels usages peut-on en faire dans la vraie vie ?** ***Réponse :*** *C'est une sécurité contre les erreurs humaines (ex : suppression accidentelle d'une table). Si une erreur est commise sur le Primary, on a 120 secondes pour récupérer les données sur ce nœud avant que l'erreur ne s'y propage.*

**23. Un client exige une lecture toujours à jour, même en cas de bascule. Quelles options de *ReadPreference* et *WriteConcern* recommanderiez-vous ?** ***Réponse :***

- `ReadPreference : primary` *(lecture stricte sur le maître).*
- `WriteConcern : "majority"` *(pour s'assurer que la donnée n'est confirmée que si elle est écrite sur la majorité, évitant un rollback en cas de bascule).*

**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 ?** ***Réponse :*** `w: 2` *(ou* `w: "majority"` *dans un cluster à 3 nœuds).*

**25. Un étudiant a lu depuis un Secondary et récupéré une donnée obsolète. Expliquez pourquoi et comment éviter cela.** ***Réponse :*** *La réplication est asynchrone (délai de propagation). Pour éviter cela, il faut utiliser* `ReadPreference: primary` *ou utiliser des sessions avec cohérence causale (Causal Consistency).*

**26. Montrez la commande pour vérifier quel nœud est actuellement Primary dans votre Replica Set.** ***Réponse :*** `rs.isMaster().primary` *ou simplement consulter la sortie de* `rs.status()`*.*

**27. Expliquez comment forcer une bascule manuelle du Primary sans interruption majeure.** ***Réponse :*** *Utiliser* `rs.stepDown()` *sur le Primary. Cela force une élection gracieuse : le Primary ferme proprement les connexions et devient Secondary, laissant la place à un autre.*

**28. Décrivez la procédure pour ajouter un nouveau nœud secondaire dans un ReplicaSet en fonctionnement.** ***Réponse :*** *1. Démarrer une instance* `mongod` *avec le paramètre* `-replSet`*. 2. Se connecter au Primary actuel. 3. Exécuter* `rs.add("nouveau_host:port")`*.*

**29. Quelle commande permet de retirer un nœud défectueux d'un Replica Set ?** ***Réponse :*** *`rs.remove("hostname:port")`*

**30. Comment configurer un nœud secondaire pour qu'il soit caché (non visible aux clients) ? Pourquoi ferait-on cela ?** ***Réponse :*** *Configuration : `hidden: true` et `priority: 0`. Usage : Pour avoir un nœud dédié aux backups ou aux analyses (reporting) sans perturber le trafic client.*

**31. Montrez comment modifier la priorité d'un nœud afin qu'il devienne le Primary préféré.** ***Réponse :*** *On récupère la config (`cfg = rs.conf()`), on modifie la priorité (`cfg.members[n].priority = 2`), puis on reconfigure (`rs.reconfig(cfg)`).*

**32. Expliquez comment vérifier le délai de réplication d'un Secondary par rapport au Primary.** ***Réponse :*** *Utiliser la commande `rs.printSecondaryReplicationInfo()` qui affiche le retard (lag) temporel.*

**33. Que fait la commande `rs.stepDown()` et dans quel scénario est-elle utile ?** ***Réponse :*** *Elle force le Primary à démissionner. Utile pour la maintenance serveur (mise à jour) ou pour tester la résilience.*

**34. Comment redémarrer un Replica Set sans perdre la configuration ?** ***Réponse :*** *Il suffit de redémarrer les processus. La configuration est persistée dans la base de données interne `local` sur le disque.*

**35. Expliquez comment surveiller en temps réel la réplication via les logs MongoDB ou commandes shell.** ***Réponse :*** *Shell : `rs.status()`. Logs : Surveiller les messages de "heartbeat" et les transitions d'état.*

## 5 Questions complémentaires

**37. Qu'est-ce qu'un Arbitre (Arbiter) et pourquoi ne stocke-t-il pas de données ?** ***Réponse :*** *Un arbitre est un membre votant qui ne possède pas de copie des données. Il sert uniquement à départager les votes pour atteindre le quorum sans consommer de ressources disque.*

**38. Comment vérifier la latence de réplication entre le Primary et les Secondaries ?** ***Réponse :*** *Dans `rs.status()`, consulter la section `members` et les champs `pingMs` et `optimeDate`.*

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

**40. Quelle est la différence entre la réplication asynchrone et synchrone ? Quel type utilise MongoDB ?** ***Réponse :*** *Synchrone : le client attend l'écriture sur tous les nœuds (lent, sûr). Asynchrone : le client n'attend que le maître (rapide). MongoDB utilise la réplication* ***asynchrone*** *par défaut.*

**41. Peut-on modifier la configuration d'un Replica Set sans redémarrer les serveurs ?** ***Réponse :*** *Oui, via la commande `rs.reconfig()`.*

**42. Que se passe-t-il si un nœud Secondary est en retard de plusieurs minutes ?** ***Réponse :*** *Il ne peut pas être élu Primary en cas de panne (données trop vieilles). S'il tombe trop loin derrière (taille de l'Oplog dépassée), il devra resynchroniser toutes les données (Initial Sync).*

**43. Comment MongoDB gère-t-il les conflits de données lors de la réplication ?** ***Réponse :*** *Le Primary a toujours raison. En cas de divergence après une partition, les écritures non répliquées de l'ancien maître sont annulées (Rollback) pour s'aligner sur le nouveau maître.*

**44. Est-il possible d'avoir plusieurs Primarys simultanément dans un Replica Set ? Pourquoi ?** ***Réponse :*** *Non, c'est un "Split-Brain". Le système l'empêche via la règle de majorité pour éviter la corruption de données.*

**45. Pourquoi est-il déconseillé d'utiliser un Secondary pour des opérations d'écriture même en lecture préférée secondaire ?** ***Réponse :*** *Ce n'est pas seulement déconseillé, c'est impossible. Un nœud Secondary rejette techniquement toute opération d'écriture.*

**46. Quelles sont les conséquences d'un réseau instable sur un Replica Set ?** ***Réponse :*** *Des élections fréquentes ("flapping"), des indisponibilités temporaires en écriture et un retard de réplication accru.*