



Chapitre 2 : Introduction à la Base de Données Orientée Documents MongoDB

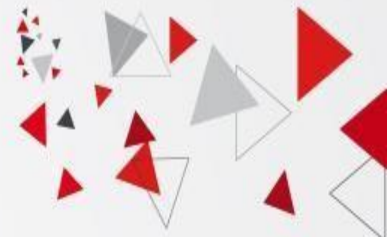
Equipe BD NoSQL



Plan

- Introduction
- Structure de données
- Terminologie
- Modélisation
- Sharding
- Replication
- Exemple

► Introduction



- Base de données NoSQL orientée documents
- Schéma flexible
- Ecrite en C++
- Développée en 2007 par la firme MongoDB





► Structure de données – document

- Données avec un schéma flexible
- Stockées sur le disque sous forme de documents BSON
 - ✓ Documents BSON (Binary JSON) : représentation binaire sérialisées d'un document JSON
 - ✓ Supporte plus de types de données que JSON (documents, tableaux, tableaux de documents,...)



Structure de données – document

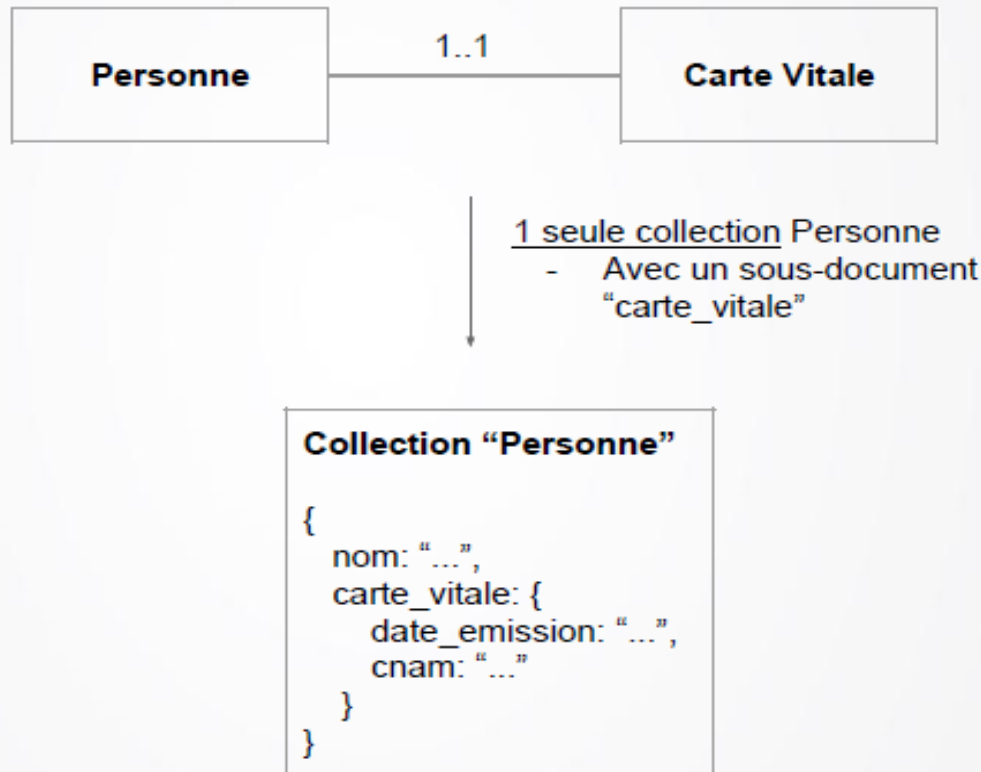
- Taille max d'un document : 16 Mo
 - ✓ Utilisation de l'API GridFS pour stocker des documents plus larges que la taille autorisée
 - ✓ GridFS permet de diviser les documents en chunks de même taille, et les stocke sous forme de documents séparés
- Les documents sont organisés en collections
- Exemples
 - ✓ {nom: "Alan", tél: 2157786, email: "agb@abc.com"}
 - ✓ {nom: {prénom: "Alan", famille: "Black"},tél: 2157786, email: "agb@abc.com"}
 - ✓ {name: "Alan", téls: [2157786, 2498762] }



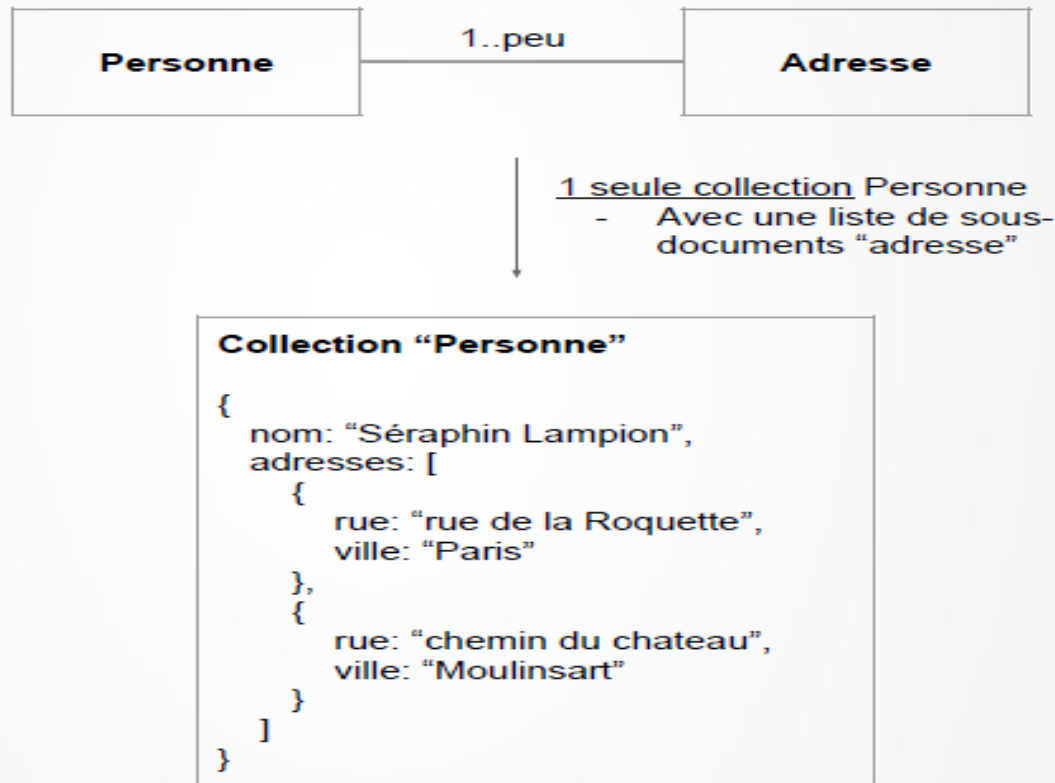
Terminologie

Base de données relationnelle	MongoDB
Base de données	Base de données
Table	Collection
Ligne	Document
Colonne	Champ
Index	Index
Jointure	Imbrication
Clé étrangère	Référence
Clé primaire	Clé primaire (représentée par le champ <code>_id</code>)

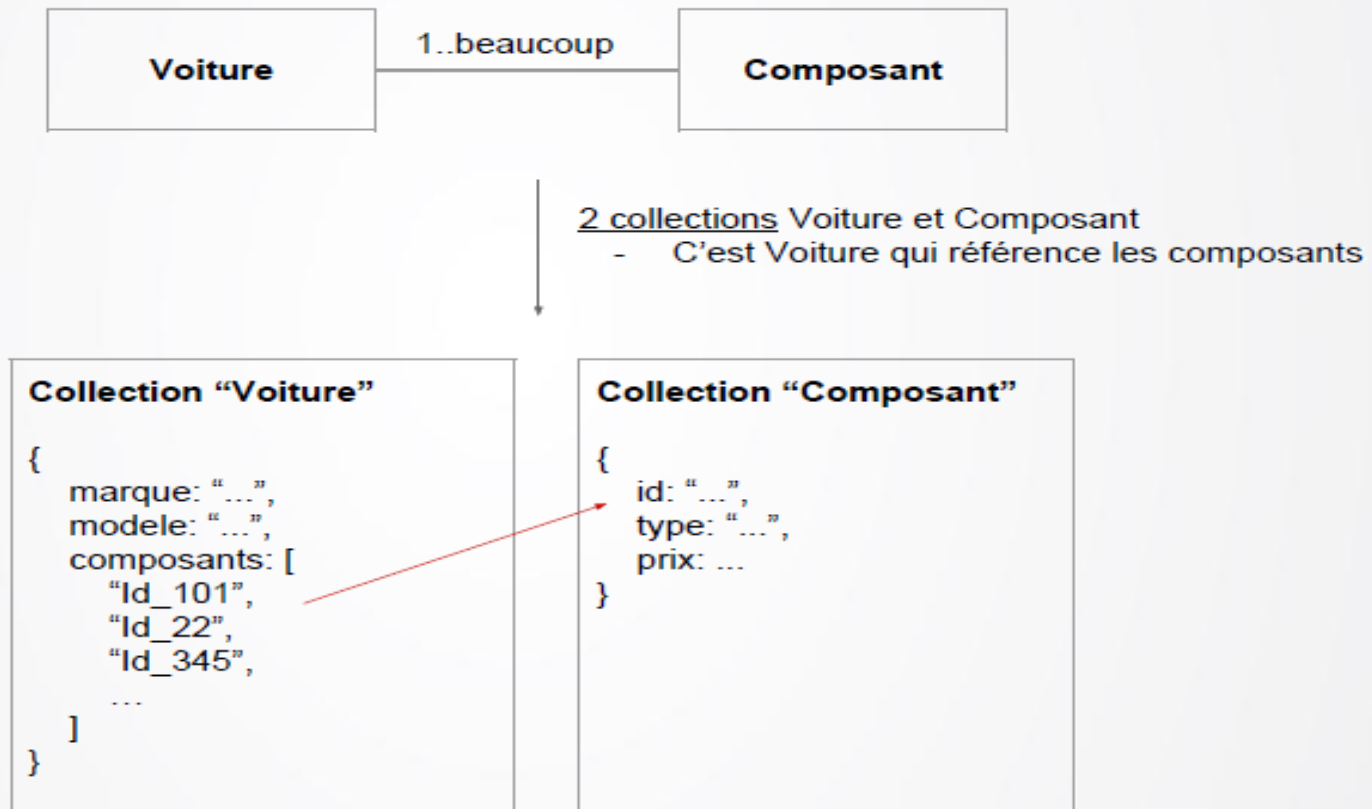
► Modélisation - quelques règles simples



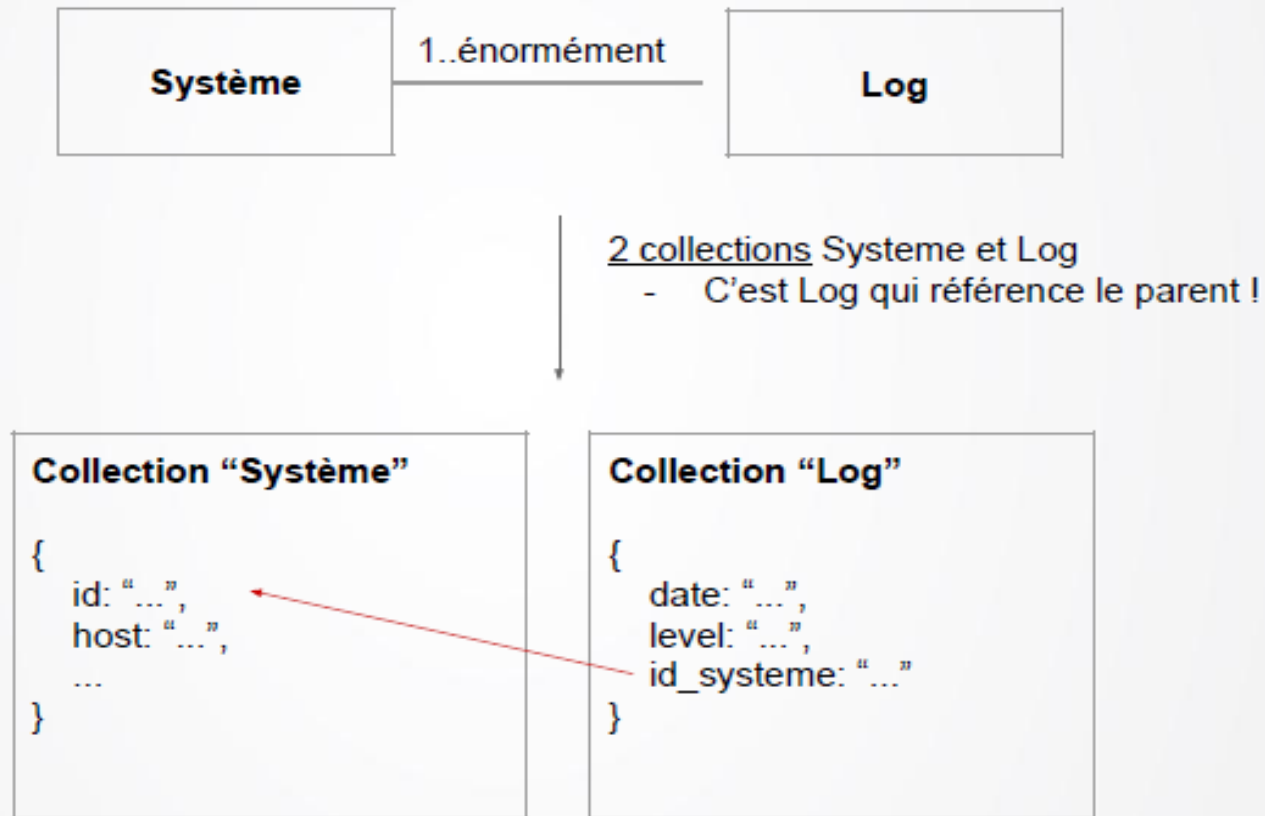
► Modélisation - quelques règles simples



► Modélisation - quelques règles simples




► Modélisation - quelques règles simples





► Modélisation - quelques règles simples

- Ne pas oubliez d'en tenir compte
- Besoin de l'ensemble des données à chaque requête
 - Une seule collection
- Besoin d'avoir seulement une partie de données
 - Plusieurs collections et des références
- Exemple : les posts d'un blog et leurs commentaires
 - ✓ 2 besoins : affichage liste des posts + affichage post avec commentaires
 - ✓ Modélisation avec 2 collections (posts, comments)

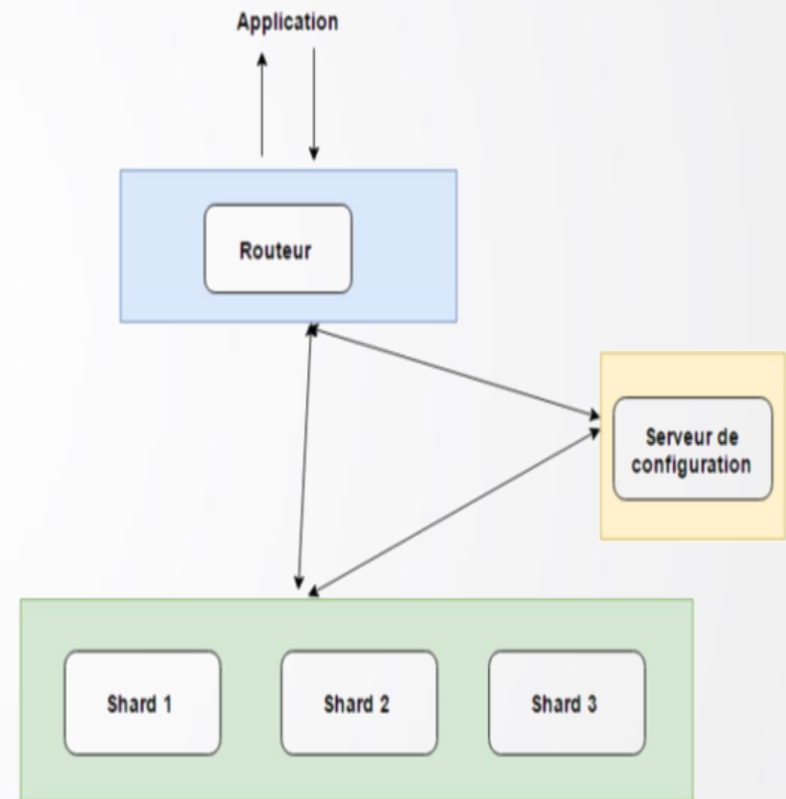


Sharding – principe

- Pour faire de la scalabilité (ou extensibilité) horizontale avec MongoDB on utilise le sharding (ou partitionnement)
- Méthode proposée par MongoDB qui permet de distribuer les données sur plusieurs machines
- Créer un cluster MongoDB (sharded cluster) composé de plusieurs machines (nœuds) sur lesquelles les données vont être réparties.
- La répartition peut être effectuée de façon arbitraire ou bien selon un `sharding_key` (ou clé de partitionnement)
- On définit comme clé un champ présent dans tous les documents.

Sharding – Architecture d'un cluster (1/4)

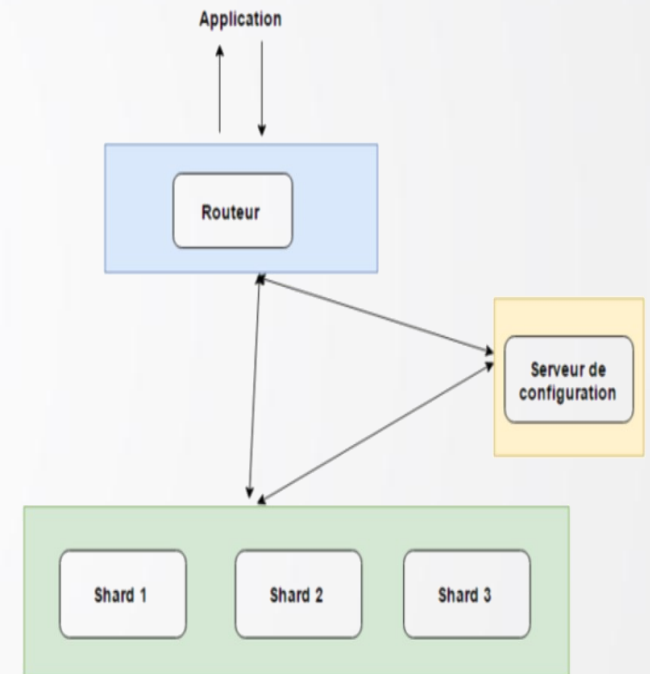
- Un sharded cluster est composé de 3 principaux éléments :
 - Serveur de configuration
 - Shards (ou nœuds)
 - Routeur



Cluster MongoDB

Sharding – Architecture d'un cluster (2/4)

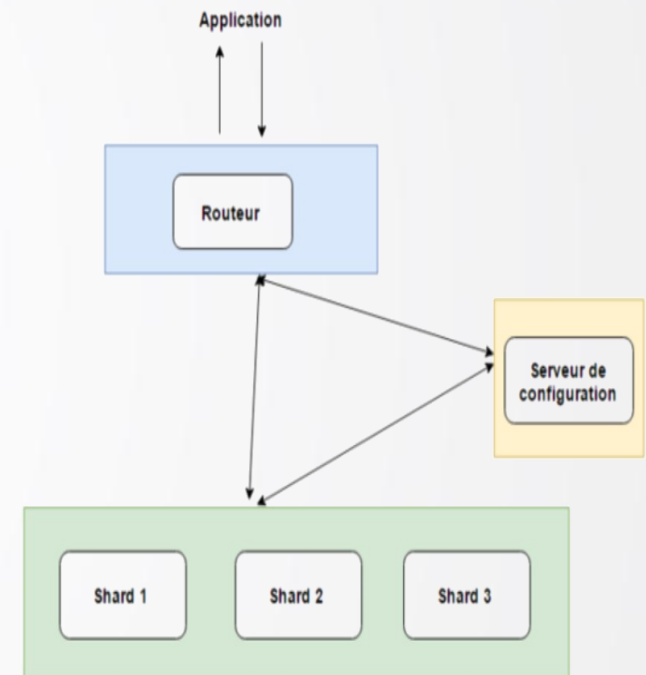
- Serveur de configuration :
 - Stocke les métadonnées et les paramètres de configuration du cluster
 - Est en charge de la localisation des données, il sait quelles données se trouvent sur quels shards
 - Agit comme un équilibreur de charge (*load balancer*)



Cluster MongoDB

Sharding – Architecture d'un cluster (3/4)

- Shard (ou nœud) :
 - Contient un sous ensemble de données
 - S'il est saturé, il suffit d'ajouter d'autres shards => scalabilité horizontale

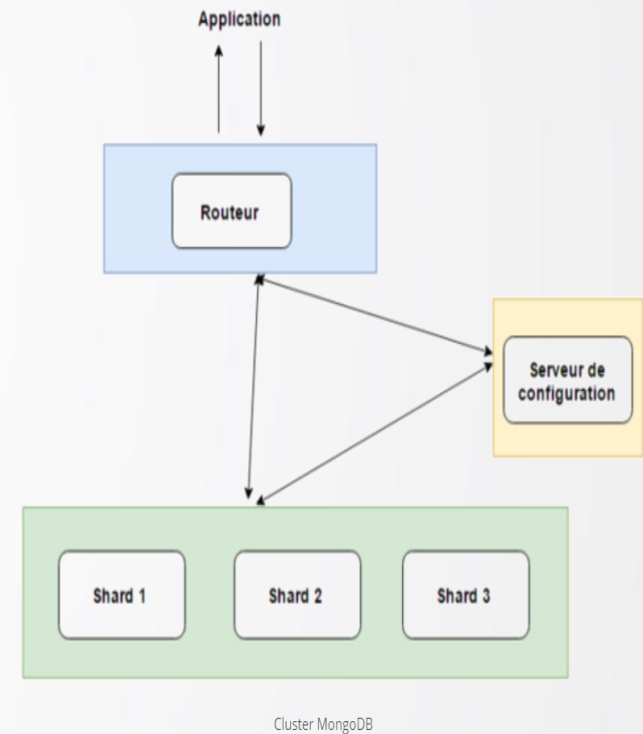


Cluster MongoDB

► Sharding – Architecture d'un cluster (4/4)

- Routeur : **mongos**

- Une instance mongos permet de router les requêtes vers le shard approprié
- Elle agit comme routeur
- Elle joue le rôle d'interface entre l'application cliente et le sharded cluster : le routeur communique avec le serveur de configuration pour connaître la répartition des données et donc choisir le bon shard.





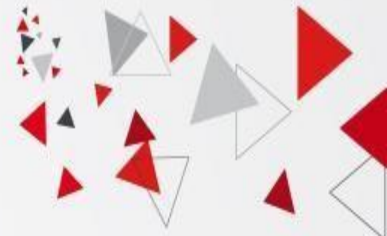
Sharding – Atouts d'un cluster

- Load balancing (répartition de charge)
- Temps de réponse plus rapides (requêtes parallélisées sur de plus petits jeux de données)
- Ajout de serveurs supplémentaires sans interruption du service

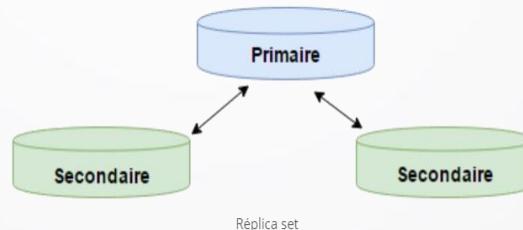
L'architecture précédente (page 13) permet de répartir la charge entre différentes machines (load balancing), mais elle n'est pas hautement disponible. Si un des serveurs tombe, le cluster ne pourra plus fonctionner.

En cas de panne, comment assurer la haute disponibilité des données et la tolérance aux pannes ?

► Replica set – Principe

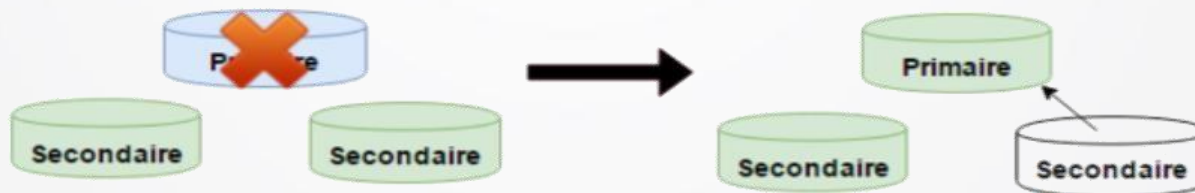


- Un sharded cluster se doit d'être hautement disponible
 - MongoDB propose un mécanisme de réplication, Le réplica set, pour se prémunir contre la perte partielle de données et assurer la continuité du service
- Un réplica set est un groupe d'instances qui maintiennent le même ensemble de données.
- Il se compose d'un nœud primaire (master) et de nœuds secondaires (slaves)
- Les opérations ont lieu sur le nœud primaire, puis elles sont reproduites sur les nœuds secondaires.

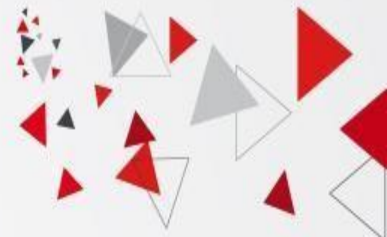


► Replica set – Election (1/2)

- Dans le cas où le nœud primaire ne répond pas pendant 10 secondes, un nœud secondaire est élu comme nouveau nœud primaire.
- Ce processus est rapide (environ une minute), automatique et complètement transparent pour l'utilisateur.
- La promotion du nœud secondaire en nœud primaire rétablit l'accès aux données.
- Suite à la reprise de l'ancien nœud primaire, il essaye de synchroniser avec le nouveau nœud primaire. S'il trouve qu'il a des données en plus il doit effectuer un Rollback pour qu'il soit à niveau avec les autres nœuds.



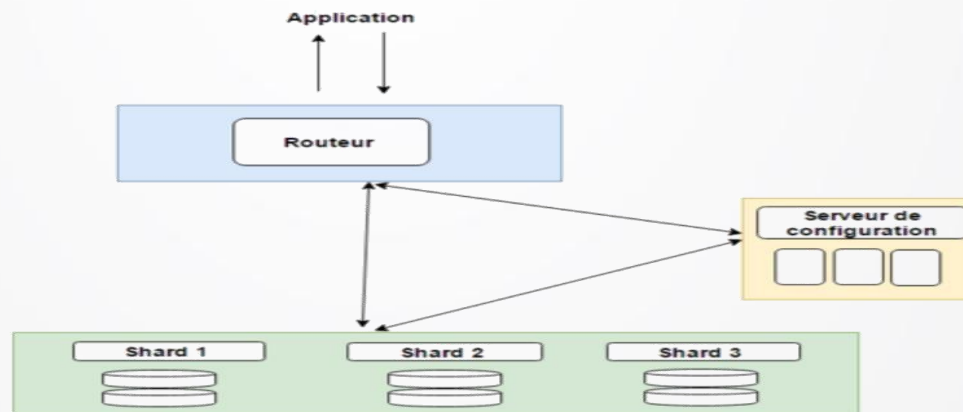
▶ Replica set – Election (2/2)



- Un replica set peut avoir jusqu'à 50 nœuds mais seulement 7 peuvent voter.
- Pour qu'un nœud soit élu nœud primaire, il doit obtenir une majorité qualifiée.
- Par exemple dans un réplica composé de 3 nœuds, pour être élu il faut au moins 2 votes.
- Si 2 serveurs tombent en panne, le nœud restant n'aura qu'un seul vote, il ne pourra donc pas être élu. Les données ne seront plus accessibles.
- Pour ne pas se retrouver dans une situation d'élection impossible, il faut ajouter un arbitre à notre réplica set.
- Un arbitre est un nœud qui ne contient pas de données et qui consomme très peu de ram. Il sert uniquement à s'assurer qu'un des nœuds aura la majorité qualifiée lors de l'élection. Il ne peut pas devenir un nœud primaire.

► Architecture hautement disponible

- On modifie l'architecture précédente (page 13) pour la rendre hautement disponible.
- Le serveur de configuration est transformé en un réplica set composé de 3 instances. Chaque shard est un réplica set formé de 2 instances et d'un arbitre.

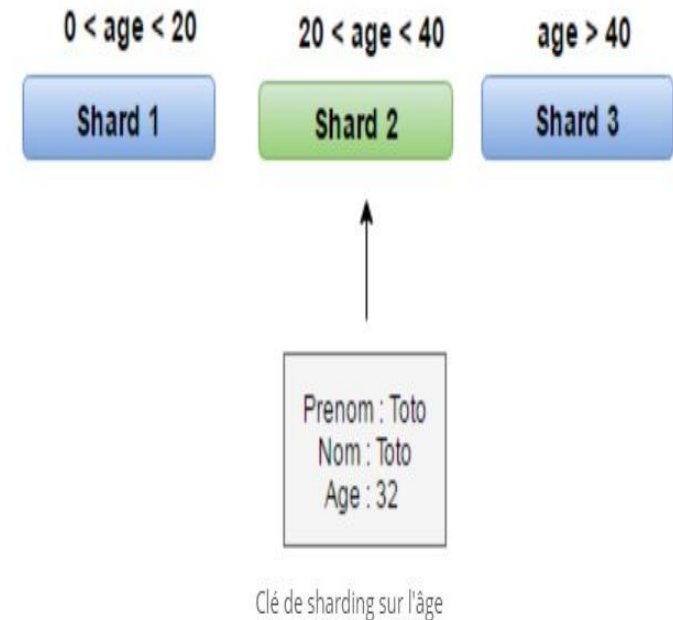


▶ Exemple (1/2)

Prenons un document `Personne` qui contient le nom, le prénom et l'âge d'une personne.

On choisit l'âge comme clé de sharding. MongoDB va définir automatiquement des intervalles d'âges. Le premier shard contiendra toutes les personnes de moins de 20 ans, le shard 2 toutes celles qui ont entre 20 et 40 ans, et le dernier shard les personnes de plus de 40 ans.

Quand un nouveau document sera ajouté dans la base, il sera directement dirigé vers le shard qui lui correspond, ici la personne a 32 ans, elle sera donc stockée sur le shard 2. C'est le routeur (mongos) qui est chargé d'orienter le document.

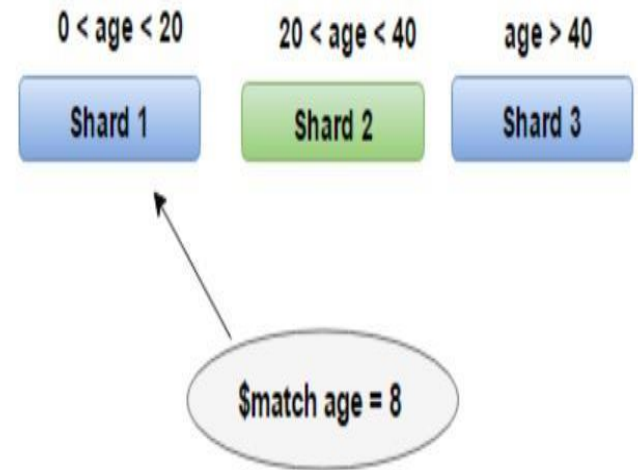


► Exemple (2/2)

Une requête portant sur la clé de sharding sera très efficace car elle ne portera pas sur l'ensemble des données, mais seulement sur le shard correspondant à la requête, c'est à dire sur un sous ensemble de données.

Dans la requêtes ci-dessus on veut connaître toutes les personnes âgées de 8 ans. Il est inutile d'interroger l'ensemble des données. Le routeur va envoyer la requête uniquement sur le premier shard qui contient les personnes de moins de 20 ans. La quantité de données sondé étant plus petite, le temps de réponse sera beaucoup plus court.

Si la requête s'applique sur plusieurs shards, elle s'effectue en parallèle sur les différents shards.



Requête sur la clé de sharding



Sources

Sites (dernières consultations mars 2020)

- Documentation officielle MongoDB : <https://docs.mongodb.com/manual/>
- Claudia Brouche, MogoDB et le sharding : <https://www.supinfo.com/articles/single/4761-mongodb-sharding>
- [Lilia Sfaxi, Bases de Données NOSQL](https://liliasfaxi.wixsite.com/liliasfaxi/big-data), <https://liliasfaxi.wixsite.com/liliasfaxi/big-data>

Présentations

- Ines Slimen, « MongoDB » , 2017

Livres Blancs

- Top 5 Considerations when evaluating NOSQL Databases, MongoDB White Paper, Juin 2018