TP MONGODB - INDEXATION, GÉOSPATIAL ET AGRÉGATION

Sans Index :

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Requête | |  | | --- | | **Documents examinés (totalDocsExamined)** |  |  | | --- | |  | | |  | | --- | | **Temps d'exécution (executionTimeMillis)** |  |  | | --- | |  | | | **Type d'étape utilisée** | | --- |  |  | | --- | |  | |
| Recherche par titre exact | 2006 | 1 | COLLSCAN |
| |  | | --- | | **Recherche par auteur** |  |  | | --- | |  | | 2006 | 1 | COLLSCAN |
| Recherche par plage de prix et note minimale | 2006 | 1 | COLLSCAN |
| Recherche filtrée par genre et langue avec tri | 2006 | 1 | COLLSCAN |

Avec Index

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Requête | |  | | --- | | **Documents examinés (totalDocsExamined)** |  |  | | --- | |  | | |  | | --- | | **Temps d'exécution (executionTimeMillis)** |  |  | | --- | |  | | | **Type d'étape utilisée** | | --- |  |  | | --- | |  | |
| Recherche par titre exact | 1 | 0 | IXSCAN |
| |  | | --- | | **Recherche par auteur** |  |  | | --- | |  | | 1 | 0 | IXSCAN |
| Recherche par plage de prix et note minimale | 405 | 1 | IXSCAN |
| Recherche filtrée par genre et langue avec tri | 551 | 0 | IXSCAN |

On peut observer que l’index permet d’effectuer des recherche précis et plus rapide comme pour la recherche du titre .On peut voir qu’il a examiné que 1 doc compare 2006 avec la méthode COOLSCAN   
  
  
  
1. Améliorations de performance observées après l'ajout d'index

Après l'ajout d'index, nous avons constaté des améliorations significatives dans le temps d'exécution des requêtes. Avant l’indexation, certaines requêtes effectuaient un COLLSCAN (lecture complète de la collection), ce qui entraînait des temps d’exécution élevés, notamment pour les recherches sur de grands ensembles de données. Avec les index, MongoDB a pu effectuer des IXSCAN (scan d’index) et limiter le nombre de documents examinés, réduisant ainsi considérablement le temps d’exécution.

2. Types d’index les plus efficaces

Les index les plus efficaces ont été :

Index simple sur les champs très utilisés pour les filtres (titre, auteur, prix, note\_moyenne).

Index composite pour des requêtes combinant plusieurs critères (ex: prix et note\_moyenne).

Index textuel sur titre et description, permettant des recherches rapides sur les mots-clés.

Index géospatial (2dsphere) pour optimiser les recherches de proximité sur les utilisateurs et bibliothèques.

L’utilisation d’index composites a été particulièrement utile pour éviter les lectures inutiles et améliorer les performances des requêtes filtrant plusieurs champs.

3. Compromis entre performance de lecture et d'écriture

L'ajout d’index améliore les performances de lecture, mais introduit des coûts en écriture :

Insertion plus lente : Lorsqu’un nouveau document est ajouté, MongoDB doit mettre à jour les index, ce qui peut ralentir les insertions massives.

Mises à jour plus coûteuses : Si des champs indexés sont modifiés fréquemment, cela entraîne une surcharge supplémentaire.

Utilisation accrue du stockage : Plus d’index signifie une base de données plus volumineuse.

Il faut donc trouver un équilibre entre vitesse de lecture et coût des opérations d’écriture.

4. Choix des index pour une application de bibliothèque en production

Pour une application réelle, nous adopterions les index suivants :

Index simple sur titre, auteur et ISBN pour des recherches rapides.

Index composite (prix, note\_moyenne) pour optimiser les recherches par prix et qualité.

Index textuel (titre, description) pour la recherche en texte intégral.

Index géospatial (localisation) pour les recherches de proximité sur les bibliothèques et utilisateurs.

Index TTL sur les sessions utilisateurs pour supprimer automatiquement les données expirées.

Enfin, nous surveillerions régulièrement l’utilisation des index avec le profiler MongoDB et supprimerions les index inutilisés pour éviter une surcharge inutile.