R403 - NoSQL

TD 5-6

DUT info Vannes

G. Kerbellec - 2022

- L'ensemble des requêtes et/ou scripts réalisées est à déposer sur moodle

1- Ressources

MongoDB permet des traitements rapides et sans contraintes sur des données massives.

Nous allons à présent voir comment il est tout de même possible de structurer l'information :

https://docs.mongodb.com/manual/core/data-modeling-introduction/

Ainsi nous pourrons comparer nos schémas aux classiques bases de données relationnelles.

Enfin, nous verrons comment utiliser les mécanismes d'agrégation pour requêter les bases de MongoDB.

2- Magasin de sport

Il vous est demandé de construire une base de données MongoDB permettant de gérer un magasin de sport.

Vous devrez dans un premier temps être capable de gérer une collection d'articles :

- marque
- prix
- nom
- référence
- catégorie (enfant, junior, senior)
- liste de tailles

- fournisseur (nom, ville)
- identifiant du rayon

Votre implémentation sera au maximum dénormalisée (informations répétées dans les documents). Ces relations sont dites “embedded”.

Afin d’introduire une forme de typage, vous créerez vos collections en utilisant les schémas ($jsonSchema) de MongoDB.

Vous pouvez vous aider du lien suivant pour produire vos schémas :

https://docs.mongodb.com/manual/core/schema-validation/

2- Normalisation ?

Un des problèmes posés par la dénormalisation est la mise à jour de données dont les clés sont réparties de manières répétées.

Il vous est demandé de pouvoir mettre à jour des informations dupliquées en utilisant la fonction updateMany.

https://docs.mongodb.com/manual/reference/method/db.collection.updateMany/#db.collection.updateMany

Vous créerez des requêtes pour :

- changer “enfant” qui devient “senior” dans la catégorie d’un article
- tentez de changer la catégorie “junior” en “loisir”
- mettre à jour l’adresse du fournisseur

A la manière des bases de données relationnelles, il est possible d’introduire de la normalisation dans MongoDB. Il suffit de créer dans un document une entrée de référence vers un autre document.

Dans votre base vous avez déjà un numéro d’identification de rayon associé à un article.

1- Créez la collection *Rayon* qui contient les informations sur sa description, l’employé responsable et dont l’_id sera celui que l’on renseigne dans les articles.

2- Ecrire un script en Node.js permettant de réaliser une jointure en retrouvant l’employé responsable d’un article donné en paramètre.

3- MongoDB a-t’il un mécanisme simple de création de jointure ? Vous pouvez tout de

même utiliser la fonction aggregate et l'exemple suivant pour la réaliser :
https://www.quackit.com/mongodb/tutorial/mongodb_create_a_relationship.cfm

3- Opérations d'agrégations

De même que SQL propose l'application des opérations (calculs, tris, etc) aux résultats de requêtes, MongoDB possède fonctions et mécanismes d'agrégations.

opérations simples

Ecrivez en utilisant les opérateurs count et distinct les requêtes suivantes :

- nombre d'articles de marque "nike"
- lister (sans doublons) les responsables de rayons

pipeline d'agrégations

L'algorithme map-reduce est un modèle de programmation qui permet des calculs de manière répartie. Il s'agit de grouper les données données puis d'y appliquer une opération (potentiellement sur différentes machines).

MongoDB introduit la notion de pipeline d'agrégations qui a les mêmes objectifs, mais est plus efficace.

En vous inspirant de l'exemple de la documentation, réalisez un pipeline d'agrégation permettant de compter les articles de la marque "nike" dans chaque rayon :

https://docs.mongodb.com/manual/core/aggregation-pipeline

Assurez-vous de compléter votre base de manière à avoir pour chaque article de type chaussure, un tableau listant les tailles et les quantités disponibles. Puis, en vous inspirant de l'exemple suivant, comptez les chaussures "senior" de taille 45 :

https://dzone.com/articles/mongodb-pipelines-with-examples

4- Exercice

A présent, il vous est demandé de gérer les achats. Modifiez et apportez les modélisations

/ données / scripts suffisants pour réaliser des ajouts d’articles ainsi que pour facturer les achats des clients tout en assurant la mise à jour des quantités.