Exercices bases de données NoSQL

TLC course

Nous verrons comment intéragir avec la base de données Google Datastore en Java

---

Google Datastore est une base de données sans schéma et NoSQL. Elle a été conçue en premier lieu pour passer à l'échelle en distribuant les données, limitant le nombre de requêtes possibles. Il est, par exemple, impossible de faire des jointures, de réaliser un filtre d'inégalités sur plusieurs propriétés ou de filtrer des données récupérées via une sous-requête.

Le Genre (Kind) est le plus large concept manipulable dans Google Datastore. Un Genre contient plusieurs objets appelés Entités (Entities). Ces Entités sont composées de Propriétés (Property) ainsi que d'un identifiant unique appelé Clé (Key). De plus Google Datastore supporte une notion de hiérarchie : une entité peut avoir un parent. Cette hiérarchie s'appelle un groupe d'entité.

Les requêtes vers les clés et les ancêtres sont fortement cohérentes (strong consistency), les autres sont cohérentes à terme (eventual consistency). Toutes les requêtes sont réalisées à partir d'index préalablement calculés. Une requête incluant des contraintes sur plusieurs propriétés nécessitera la création d'un index composite par le développeur. Il est possible d'envoyer des requêtes par lot (batch) dans Google Datastore pour accélérer leur traitement. Ces requêtes par lot ne sont pas à confondre avec les transactions, limitées à un groupe d'entité et assurant que, soit toutes les requêtes réussissent, soit toutes les requêtes sont annulées.

La création des clés peut être déléguée à Google Datastore ou être gérée par l'application selon des règles propres au développeur.

Question 1 : Faites un parallèle avec la sémantique SQL traditionnelle. À quoi correspond un Genre, une Entité, etc. Quelles sont les différences également ?

Question 2 : Certaines requêtes sont cohérentes à terme. À quoi devez-vous faire attention dans votre application ? Vous pouvez prendre l'exemple d'une application bancaire autorisant des virements entre utilisateurs.

Question 3 : Proposez une organisation des données pour le stockage de messages d'une plateforme de micro-blogging autorisant les hashtags. Les données à stocker à minima sont le contenu du message, le nom de l'utilisateur et lieu. Proposez également des index composites afin de pouvoir chercher avec n'importe quel combinaison de hashtag, utilisateur et lieu.

Question 4 : Donnez un exemple où vous pourriez tirer parti des requêtes par lot et des transactions dans le cas de l'application de micro-blogging évoquée précédemment.

Question 5 : Dans le cadre de la RGPD, vous devez respecter le droit à l'effacement dans votre application de micro-blogging. Vous devez donc être en mesure de supprimer tous les messages d'un utilisateur. Hors, l'API de Google Datastore ne supporte que la suppression par clé. Comment allez vous faire ? Pourrez-vous utiliser une opération par lot ou des transactions pour accélérer le processus ?

Question 6 : Quels sont les avantages et inconvénients de laisser Google Datastore gérer les clés de mon application ? et réciproquement, de les gérer moi même ?

Exemple d'utilisation de Google Datastore en Java :

```
// Setup
Datastore datastore = DatastoreOptions.getDefaultInstance().getService();
KeyFactory recordsKey = datastore.newKeyFactory().setKind("content");

// Insert a new entity
Key recKey = datastore.allocateId(recordsKey.newKey());
Entity record = Entity
  .newBuilder(recKey)
  .set("hello","world")
  .set("foo","bar")
  .set("price", 5)
  .build();
datastore.add(record);

// Update and Get with key can be done similarly

// Query entities
Filter costly = PropertyFilter.gt("price", 3);
Filter intl = PropertyFilter.eq("hello", "world");

Query q1 = new Query("content").setFilter(costly);
Query q2 = new Query("content")
  .setFilter(CompositeFilterOperator.and(costly, intl));

for (Entity result : datastore.prepare(q2).asIterable()) {
  System.out.println((String) result.getProperty("hello"));
}

// Delete an entity
datastore.delete(recKey);

// Batch operations can be achieved by:
//  * passing a List<Entity> to datastore.add()
//  * passing a List<Key> to datastore.delete()
```

Question 7 : Donnez à chaque modification de la base, sont état. Par exemple, au début elle est vide, et n'a donc pas de Genre, ni d'Entité, etc.

Question 8 : À l'aide des exemples de code fournis et de l'exemple de la plateforme de micro-blogging, écrivez le code pour insérer un nouveau message dans la base, pour pouvoir chercher selon toutes les combinaisons de hashtags, nom d'utilisateur et lieu et supprimer tous les messages d'un utilisateur.

— the end —