Apache Solr

Orientée Dev (1 jour)

Programme Orientée Dev

- Indexation de données.
- Recherche de données
- Optimisation de la performance
- Personnalisation de Solr
- Interaction avec SGBDR

Indexation de données

- Comprendre comment utiliser l'API REST de Solr pour indexer des données
- Comprendre les options de configuration de l'indexation pour optimiser la performance et la pertinence
- Comprendre comment utiliser les outils de gestion de données de Solr, tels que DataImportHandler (DIH)

Indexation de données

Comprendre comment utiliser l'API REST de Solr pour indexer des données

- L'indexation est le processus de conversion des documents en un format qui peut être rapidement recherché par Solr
- Une des caractéristiques les plus puissantes de Solr est son API REST, qui permet d'indexer, de rechercher, de mettre à jour et de supprimer des données facilement.
- **Indexation simple**

```
curl http://localhost:8983/solr/contact/update?commit=true -d '
[
  {
    "id": "1",
    "nom": "Dupont",
    "prenom": "Jean"
  }
]'
```

Indexation de données

Comprendre comment utiliser l'API REST de Solr pour indexer des données

- **Indexation en masse**
- Au lieu d'envoyer des documents individuellement, vous pouvez envoyer plusieurs documents en une seule requête pour améliorer l'efficacité.

```
curl http://localhost:8983/solr/contact/update?commit=true -d '
[
  {"id": "1", "nom": "Dupont", "prenom": "Jean"},
  {"id": "2", "nom": "Martin", "prenom": "Pierre"},
  ...
]'
```

Indexation de données

Comprendre comment utiliser l'API REST de Solr pour indexer des données

- L'idée est d'améliorer les performances en minimisant les opérations d'écriture sur le disque, qui peuvent être coûteuses en termes de temps
- **Performance** : L'indexation en buffer peut augmenter considérablement les performances d'indexation car les opérations en mémoire sont généralement beaucoup plus rapides que les opérations d'écriture sur le disque.
- **Risques** : Bien que l'indexation en buffer soit plus rapide, elle comporte également des risques. Par exemple, si le processus est interrompu ou si la machine s'éteint avant que les données en mémoire ne soient écrites sur le disque, ces données peuvent être perdues.
- **Commit** : Dans Solr, une opération de "commit" est nécessaire pour rendre les documents indexés recherchables. Si vous effectuez des commits trop fréquemment avec un grand volume de données en buffer, cela peut ralentir le processus. Il est donc recommandé d'ajuster la fréquence des commits en fonction de votre cas d'utilisation.

Indexation de données

Comprendre comment utiliser l'API REST de Solr pour indexer des données

- **AutoCommit** : Solr offre une fonctionnalité "autoCommit" qui peut être configurée pour effectuer automatiquement des commits après un certain nombre de documents indexés ou après un certain intervalle de temps. Cela peut être utilisé pour équilibrer les besoins de fraîcheur des données et les performances d'indexation.
- **Tuning** : La taille du buffer d'indexation peut généralement être ajustée en fonction de la mémoire disponible et des besoins spécifiques de l'application. Un buffer plus grand permet d'indexer plus de documents en mémoire avant de les écrire sur le disque, mais il consomme également plus de mémoire.

Indexation de données

Comprendre comment utiliser l'API REST de Solr pour indexer des données

- Indexation de documents JSON:

```
curl -X POST -H "Content-Type: application/json" \
  --data-binary '
[
  {"id":"1", "title":"Premier document", "content":"Ceci est un exemple de document JSON."},
  {"id":"2", "title":"Deuxième document", "content":"Un autre exemple de document JSON."}
]' \
"http://localhost:8983/solr/nom_de_votre_collection/update?commit=true"
```

- Indexation de documents XML:

```
curl -X POST -H "Content-Type: application/xml" \
  --data-binary '
<add>
  <doc>
    <field name="id">3</field>
    <field name="nom">toto</field>
    <field name="prenom">tata</field>
  </doc>
</add>
' \
"http://localhost:8983/solr/nom_de_votre_collection/update?commit=true"
```

Indexation de données

Comprendre comment utiliser l'API REST de Solr pour indexer des données

- Indexation de documents csv:

```
id,nom,prenom
4,Dupont 4, Jean 4
5,Dupont 5, Jean 5
```

```
curl -X POST -H "Content-Type: application/csv" \
  --data-binary @exemple.csv \
"http://localhost:8983/solr/nom_de_votre_collection/update?commit=true"
```

- Solr Cell avec Apache Tika : Solr Cell utilise Apache Tika pour extraire et indexer le contenu de divers types de fichiers comme Word, PDF, etc.

Prérequis: Assurez-vous que le plugin Solr Cell est inclus avec votre distribution Solr et que votre configuration Solr prend en charge l'indexation de documents

```
curl "http://localhost:8983/solr/nom_de_votre_collection/update/extract?commit=true" \
  -F "myfile=@chemin/vers/votre/document.pdf" \
  -F "literal.id=doc1" \
  -F "uprefix=attr_" \
  -F "fmap.content=attr_content"
```

Indexation de données

Comprendre comment utiliser l'API REST de Solr pour indexer des données

- Indexation en temps réel (Real-Time Get): L'indexation en temps réel dans Solr vous permet de récupérer un document juste après l'avoir indexé sans avoir à attendre un commit

```
curl -X POST -H "Content-Type: application/json" \
  --data-binary '
[
  {"id":"1001", "title":"Document en temps réel", "content":"Ceci est un exemple de document pour le Real-Time Get."}
]' \
"http://localhost:8983/solr/nom_de_votre_collection/update"
```

```
curl "http://localhost:8983/solr/nom_de_votre_collection/get?id=1001&realtime=true"
```

Indexation de données

Comprendre les options de configuration de l'indexation pour optimiser la performance et la pertinence

1. **Schéma**.
2. **Commit**.
3. **Tokenizers & Filters**.
4. **Merge Policy**: Solr permet de configurer la manière dont les segments d'index sont fusionnés. Les bonnes politiques de fusion peuvent améliorer les performances d'indexation.
5. **DocValues**: Si vous avez des champs que vous utilisez fréquemment pour le tri, le faceting ou les statistiques, considérez l'utilisation de DocValues pour améliorer les performances

Indexation de données

Comprendre comment utiliser les outils de gestion de données de Solr : DataImportHandler (DIH)

- Le **DataImportHandler (DIH)** est un plugin intégré de Solr qui permet d'indexer du contenu provenant d'une base de données, d'un flux XML, de fichiers CSV et de nombreuses autres sources.

```
<requestHandler name="/dataimport" class="solr.DataImportHandler">
  <lst name="defaults">
    <str name="config">data-config.xml</str>
  </lst>
</requestHandler>
```

```
<dataConfig>
  <dataSource type="JdbcDataSource"
              driver="com.mysql.jdbc.Driver"
              url="jdbc:mysql://localhost/dbname"
              user="dbuser"
              password="dbpassword" />
  <document>
    <entity name="table1" query="SELECT * FROM table1">
      <field column="db_column1" name="solr_field1" />
      <field column="db_column2" name="solr_field2" />
      <!-- Autres champs... -->
    </entity>
  </document>
</dataConfig>
```

Indexation de données

Comprendre comment utiliser les outils de gestion de données de Solr : DataImportHandler (DIH)

- Commande de démarrage:

```
curl "http://localhost:8983/solr/nom_de_votre_collection/dataimport?command=full-import"
```

- Vérification du statut:

```
curl "http://localhost:8983/solr/nom_de_votre_collection/dataimport?command=status"
```

- DIH permet également des importations delta pour indexer uniquement les modifications depuis le dernier import.
- Il prend en charge de nombreux processeurs de transformation pour manipuler des données lors de l'indexation.
- Vous pouvez combiner plusieurs sources de données, exécuter des transformations, etc

Indexation de données

Comprendre comment utiliser les outils de gestion de données de Solr : DataImportHandler (DIH)

- **Configuration pour l'importation CSV**

```
<dataConfig>
  <dataSource type="FileDataSource" encoding="UTF-8" />
  <document>
    <entity name="file" processor="FileListEntityProcessor" baseDir="/path/to/csv/directory" fileName=".*\.csv" rootEntity="false" transformer="TemplateTransformer,CsvTransformer"
    header="true">
      <field column="column1" name="solr_field1" />
      <field column="column2" name="solr_field2" />
      <!-- Autres champs... -->
    </entity>
  </document>
</dataConfig>
```

- **Importation delta**

```
<entity name="table1"
        query="SELECT * FROM table1"
        deltaImportQuery="SELECT * FROM table1 WHERE id='${dataimporter.delta.id}'"
        deltaQuery="SELECT id FROM table1 WHERE last_modified > '${dataimporter.last_index_time}'">
```

```
curl "http://localhost:8983/solr/nom_de_votre_collection/dataimport?command=delta-import"
```

Indexation de données

Exercice

Vous êtes responsable de la mise en œuvre d'un nouveau moteur de recherche pour une bibliothèque en ligne utilisant Apache Solr. La bibliothèque possède une base de données contenant des informations sur les livres, y compris le titre, l'auteur, le résumé, le genre, la date de publication et quelques critiques

1. Créez une nouvelle collection dans Solr nommée "bibliothèque"
2. À l'aide de l'API REST de Solr, indexez les données suivantes pour un livre.
3. Supposons que vous ayez une base de données MySQL contenant une table "livres" avec des colonnes correspondant aux champs ci-dessus.

- Configurez le data-config.xml pour DIH afin de se connecter à cette base de données et de récupérer les informations des livres.
- Configurez une importation delta dans DIH pour n'indexer que les livres ajoutés ou modifiés depuis le dernier import.

4. Pourquoi serait-il avantageux d'utiliser DocValues pour le champ genre ?

Recherche de données

Comprendre comment utiliser l'API REST de Solr pour effectuer des recherches

- L'API REST de Solr est essentiellement un point d'entrée HTTP pour interagir avec Solr. Pour effectuer des recherches, nous utilisons généralement le point de terminaison /select.

```
curl "http://localhost:8983/solr/maCollection/select?q=*:*"
```

Recherche de données

Comprendre comment utiliser l'API REST de Solr pour effectuer des recherches

- **Requêtes de filtrage (Filter Queries - FQ)**: Elles permettent de filtrer les résultats retournés par une requête.

```
curl "http://localhost:8983/solr/maCollection/select?q=*:*&fq=auteur:'J.K. Rowling'"
```

- **Requêtes de tri (Sort Queries)**: Vous pouvez trier les résultats selon un ou plusieurs champs.

```
curl "http://localhost:8983/solr/maCollection/select?q=*:*&sort=date_de_publication desc"
```

- **Requêtes facettées**: Elles permettent d'obtenir des décompositions de vos résultats selon certaines catégories ou "facettes"

```
curl "http://localhost:8983/solr/maCollection/select?q=*:*&facet=true&facet.field=genre"
```

Recherche de données

Comprendre comment utiliser l'API REST de Solr pour effectuer des recherches

- **Requêtes de suggestion**:

```
curl "http://localhost:8983/solr/maCollection/suggest?q=Harry"
```

- **Requêtes géospatiales**

```
curl "http://localhost:8983/solr/maCollection/select?q=*:*&fq={!geofilt}&sfield=location&pt=40.769,-73.954&d=5"
```

Recherche de données

Comprendre comment personnaliser le score de pertinence pour les résultats de recherche

- La personnalisation du score est cruciale pour fournir des résultats pertinents à vos utilisateurs.

1. **Utilisation du paramètre boost**: Vous pouvez augmenter ou diminuer la pertinence d'un document en fonction de la valeur d'un champ

```
curl "http://localhost:8983/solr/maCollection/select?q=roman&boost=recip(ms(NOW,date_de_publication),3.16e-11,1,1)"
```

2. **Function Queries**: Cela permet d'appliquer une fonction à un champ pour influencer le score.

```
curl "http://localhost:8983/solr/maCollection/select?q=roman&boost=field(critiques)"
```

3. **Écrire votre propre Similarity**: Si vous voulez avoir un contrôle total sur la manière dont le score est calculé, vous pouvez écrire votre propre Similarit

Recherche de données

Exercice

Imaginons que vous travailliez pour une grande bibliothèque universitaire qui a récemment intégré Apache Solr pour gérer son catalogue en ligne de livres, articles, journaux et autres ressources.
Votre catalogue Solr contient les champs suivants pour chaque item :

```
id: Un identifiant unique pour l'item.
titre: Le titre de l'item.
auteur: L'auteur ou les auteurs de l'item.
genre: Le genre ou la catégorie de l'item (par exemple, "histoire", "science", "fiction").
date_de_publication: La date de publication de l'item.
localisation: Un champ de coordonnées géographiques indiquant l'emplacement physique de l'item dans la bibliothèque (si applicable).
nombre_de_citations: Le nombre de fois que cet item a été cité dans d'autres travaux.
```

1. Utilisez l'API REST de Solr pour retrouver tous les items dont le titre contient le mot "Histoire"
2. Recherchez tous les articles écrits par "Albert Einstein". Filtrez les résultats pour ne montrer que ceux publiés après 1930 et triez-les par date_de_publication en ordre décroissant
3. Combien d'items dans le catalogue appartiennent à chaque genre? Utilisez une requête facettée pour le déterminer.

Recherche de données

Exercice

4. Si un utilisateur tape le mot "relatvity", il est probable qu'il ait fait une faute de frappe pour le mot "relativity". Configurez Solr pour suggérer des corrections pour ce genre d'erreurs
5. Imaginez que vous ayez une application mobile pour la bibliothèque. Un utilisateur souhaite trouver tous les items disponibles dans une certaine section de la bibliothèque. Utilisez une requête géospatiale pour retrouver tous les items dans un rayon de 5 mètres autour d'un point donné
6. Les articles qui sont fréquemment cités sont généralement considérés comme importants ou influents. Ajustez le score de pertinence pour donner plus de poids aux items ayant un grand nombre_de_citations

Intégration avec les applications PHP

Comprendre comment intégrer Solr avec les applications PHP

- Les applications PHP peuvent utiliser Solr comme back-end de recherche pour améliorer la performance et la pertinence de leurs systèmes de recherche. Solr étant un serveur basé sur Java, il communique avec d'autres langages de programmation via HTTP. PHP peut donc interagir avec Solr grâce à ses capacités de communication HTTP et à des bibliothèques spécifiques

```php
$query = "1984"; // exemple de terme de recherche
$response = file_get_contents("http://localhost:8983/solr/ma_collection/select?q=titre:$query");
$data = json_decode($response);
print_r($data->response->docs);
```

Intégration avec les applications PHP

Comprendre comment utiliser les bibliothèques PHP

Il existe plusieurs bibliothèques PHP qui facilitent l'interaction avec Solr. Deux des plus populaires sont Solarium et SolrPHP

1. **Solarium**: Solarium est une bibliothèque client pour PHP. Elle fournit un moyen simple et flexible d'utiliser Solr dans votre application PHP.

```php
require 'vendor/autoload.php';

$config = array(
    'endpoint' => array(
        'localhost' => array(
            'host' => '127.0.0.1',
            'port' => 8983,
            'path' => '/solr/ma_collection/'
        )
    )
);

$client = new Solarium\Client($config);
$query = $client->createSelect();
$query->setQuery('titre:1984');
$resultset = $client->select($query);

foreach ($resultset as $document) {
    echo $document->titre . '<br/>';
}
```

Intégration avec les applications PHP

Comprendre comment utiliser les bibliothèques PHP

2.**SolrPHP**: Cette bibliothèque n'est pas aussi populaire que Solarium, mais elle offre également des fonctionnalités pour communiquer avec Solr

Intégration avec les applications PHP

Utilisation des fonctionnalités de Solr pour une recherche et une indexation en temps réel avec les applications PHP

- L'une des forces de Solr est son indexation en temps réel. Cela signifie que dès qu'un document est ajouté à l'index, il est immédiatement disponible pour la recherche.
- Apache Solr offre en plus une multitude de fonctionnalités qui peuvent être exploitées dans les applications PHP grâce à la communication via HTTP.

1. Faceting
2. Highlighting
3. Spell Checking
4. MoreLikeThis
5. Grouping
6. Spatial Searches
7. Advanced Filtering

Intégration avec les applications PHP

Exercice

Vous allez créer une simple application web PHP qui permet aux utilisateurs de rechercher des livres par titre, auteur ou genre à partir d'un index Solr. Vous utiliserez la bibliothèque Solarium pour communiquer avec Solr.

1. Dans votre application PHP, créez une page recherche.php.
2. Dans cette page, ajoutez un formulaire qui permet à l'utilisateur de saisir un terme de recherche et de choisir s'il souhaite rechercher par titre, auteur ou genre
3. Dans le même fichier, après la soumission du formulaire, utilisez la bibliothèque Solarium pour envoyer une requête à Solr en fonction des entrées de l'utilisateur.
4. Affichez les résultats retournés par Solr sur la page.
5. Ajoutez des fonctionnalités telles que la mise en évidence des termes recherchés dans les résultats.
6. Implémentez la pagination pour les résultats si plus de 10 éléments sont retournés.
7. Proposez des suggestions de recherche si aucun résultat n'est trouvé

Optimisation de la performance de Solr pour une utilisation en production

- L'optimisation de Solr pour un environnement de production est essentielle pour garantir des temps de réponse rapides et une utilisation efficace des ressources système
- **Soyez sélectif avec les champs retournés**: Si vous n'avez pas besoin de tous les champs d'un document, utilisez le paramètre fl pour spécifier uniquement les champs nécessaires
- **Évitez les requêtes wildcard coûteuses**: Les requêtes comme :* ou texte:*potter peuvent être coûteuses en termes de performance
- **Utilisez la pagination**: Si votre recherche renvoie des milliers de documents, utilisez les paramètres start et rows pour paginer les résultats

Personnalisation de l'interface utilisateur de Solr

- L'interface utilisateur par défaut de Solr, souvent appelée "Solr Admin UI", est principalement destinée à l'administration et au débogage. Cependant, si vous souhaitez l'adapter aux besoins spécifiques de votre entreprise

1. **Personnaliser le tableau de bord**: Vous pouvez modifier l'aspect visuel en remplaçant les fichiers CSS et les images. Ceux-ci se trouvent généralement dans le dossier webapp/web/css et webapp/web/img.
2. **Modifier les templates**: Les templates de l'interface sont des fichiers Velocity et peuvent être trouvés dans le dossier webapp/web. En modifiant ces fichiers, vous pouvez changer la disposition, ajouter des éléments supplémentaires ou supprimer des fonctionnalités.
3. **Écrire un plugin personnalisé**

Comment utiliser les bases de données relationnelles pour stocker les logs d'audit Solr

- Stocker les logs d'audit dans une base de données relationnelle nécessite généralement une intégration externe ou un log shipper
- L'une des approches courantes consiste à utiliser le pipeline "Filebeat -> Logstash -> Base de données relationnelle"
    1. Utilisez un outil comme Filebeat pour lire les logs Solr.
    2. Configurez Filebeat pour envoyer des logs à Logstash.
    3. Dans Logstash, configurez un pipeline pour analyser les logs et les envoyer à votre SGBDR.

TP

- **Contexte**: "Le Quotidien Lumière" est un journal en ligne populaire avec des dizaines de milliers d'articles couvrant divers sujets depuis sa création. Ils utilisent actuellement une simple fonction de recherche basée sur une base de données, mais veulent améliorer la pertinence et la performance de leurs recherches. Ils veulent aussi exploiter les capacités avancées de recherche offertes par Solr.

1. Importez une collection d'articles réels du journal avec différents contenus : actualités, sports, culture, technologie, etc.
2. Chaque article devrait avoir un titre, un résumé, le contenu complet, la date de publication, l'auteur, et une liste de tags
3. Effectuez une recherche simple sur un mot-clé, par exemple "sport".
4. Affichez les résultats par pertinence.
5. Mettez en place une recherche facettée basée sur les auteurs et les tags. Par exemple, combien d'articles ont été écrits par "Jean Dupont" sur le "football"?
6. Indexez les articles avec des informations géographiques (par exemple, lieux des événements reportés).

TP

7. Effectuez une recherche pour trouver tous les articles liés à un événement survenu dans un rayon de 10 km d'une certaine localité
8. Renvoyez les articles triés par date de publication.
9. Mettez en œuvre une pagination pour afficher les résultats 10 par 10.
10. Configurez Solr pour fournir des suggestions lorsqu'un utilisateur commence à taper une requête. Testez avec des requêtes comme "pol" et vérifiez si "politique" est suggéré
11. Expérimentez avec les fonctionnalités de boosting pour augmenter la pertinence des articles récents ou des articles d'auteurs populaires. Testez vos configurations en vérifiant comment les articles sont classés pour certaines requêtes