Université de Cergy-Pontoise

RAPPORT : BDA

pour le projet Atelier de Gestion de Projet

Master 1 Informatique et Ingénierie des Systèmes Complexes (IISC)

sur le sujet

Système intelligent de création d'offres de voyage

Auteurs :
Feriel MALEK
Cylia BELKACEMI
Amaury FERRY
Thomas FERNANDEZ

Encadrants :
Mr. LIU Tianxiao
Mr. VODISLAV Dan

Janvier 2024

Table des matières

Table des figures

Liste des tableaux

Remerciements

Au cours de la réalisation de ce projet, nous tenons à exprimer notre profonde gratitude envers nos professeurs, M.LIU Tianxiao et M.VODISLAV Dan, pour leur précieuse assistance et le temps qu'il nous ont généreusement consacré. Leur disponibilité à répondre à nos interrogations, Leur soutien constant et leurs conseils éclairés ont été des atouts essentiels pour la réussite de notre travail. De même, nous souhaitons également remercier chaleureusement les membres de notre groupe pour leur engagement exceptionnel et leur esprit collaboratif qui ont permis de mener à bien le projet en un temps record.
Cette expérience a été rendue significative grâce à la cohésion de notre équipe, et nous sommes reconnaissants envers chacun de ses membres.

1 Introduction

Pendant le projet "Atelier de Gestion de Projet", notre objectif était de développer une extension pour une base de données relationnelle. Cette extension devait non seulement répondre aux requêtes SQL traditionnelles, mais également être capable de traiter des requêtes impliquant du contenu textuel. Nous avons mis en place une API spécifique chargée d'extraire les données textuelles et non textuelles à partir des requêtes SQL. Ensuite, cette API relie les données générées par le moteur d'indexation Lucene et JDBC, permettant ainsi de gérer efficacement les requêtes mixtes.

2 Structure de la base de données

Cette section expose la conception de la base de données à travers deux aspects clés : le Modèle Conceptuel des Données (MCD) et le Modèle Logique des Données. Le MCD offre une vision globale des données, tandis que le Modèle Logique précise leur représentation. Enfin, le schéma relationnel concret détaille l'organisation des tables pour assurer la cohérence des données. Cette partie présente de manière concise ces éléments fondamentaux de la structure de notre base de données.

2.1 Modèle conceptuel des données (MCD)

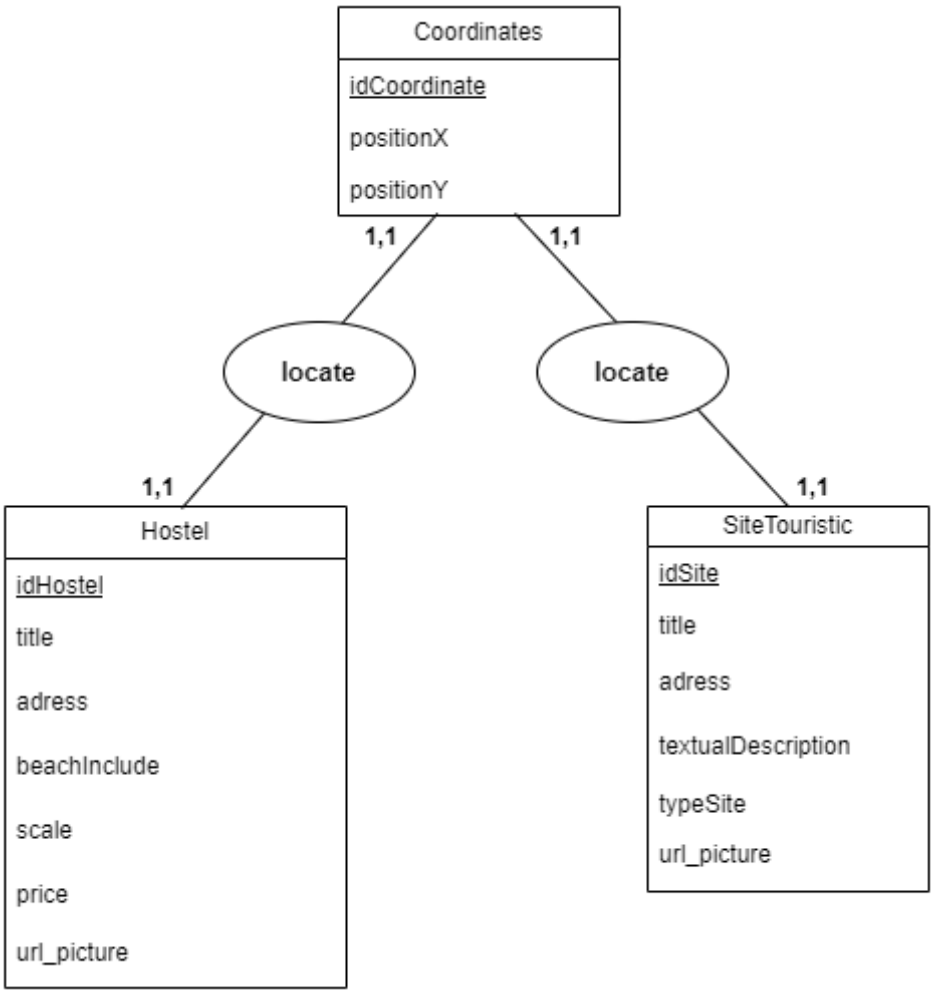

FIGURE 1 – Schéma MERISE de notre base de données

2.2 Modèle logique des données (MLD)

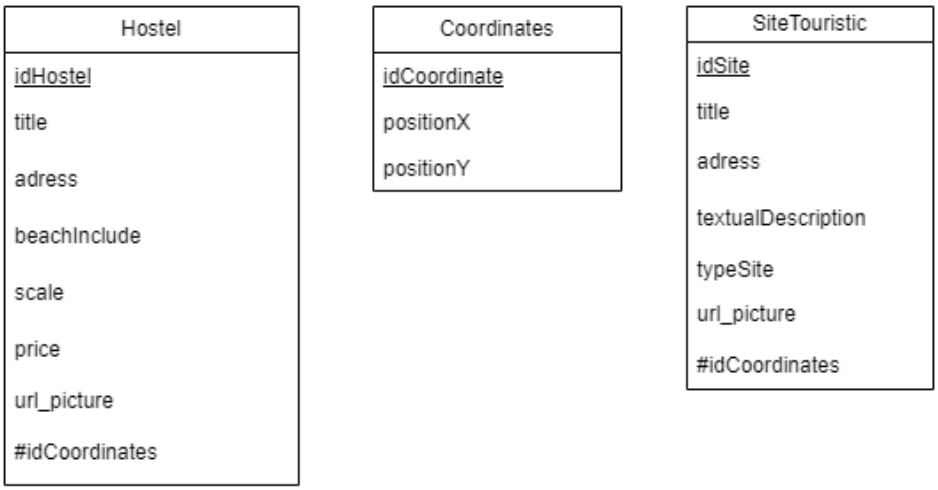

FIGURE 2 – Schéma logique de notre base de données

Notre base de données contient trois entités principales : **Coordinates, Hostel, et SiteTouristic.** Chaque Hostel et chaque SiteTouristic a une relation avec Coordinates grâce à la clé étrangère **idCoordinate**. Cela permet de localiser géographiquement ces entités sur une carte. Il y'a une association indirecte entre Hostel et SiteTouristic à travers leur relation commune avec Coordinates. Cela signifie que nous pourrions déterminer la proximité d'une hôtel à un site touristique en évaluant les distances géographiques à l'aide des coordonnées.

2.3 Schéma relationnel

Coordinates (idCoordinate, positionX, positionY)

Hostel (idHostel, name, adress, beachInclude, range, price, url_picture, *idCoordinate*)

SiteTouristic (idSite, name, adress, textualDescription, typeSite, url_picture, *idCoordinate*)

3 Conception de la base de donné étendue

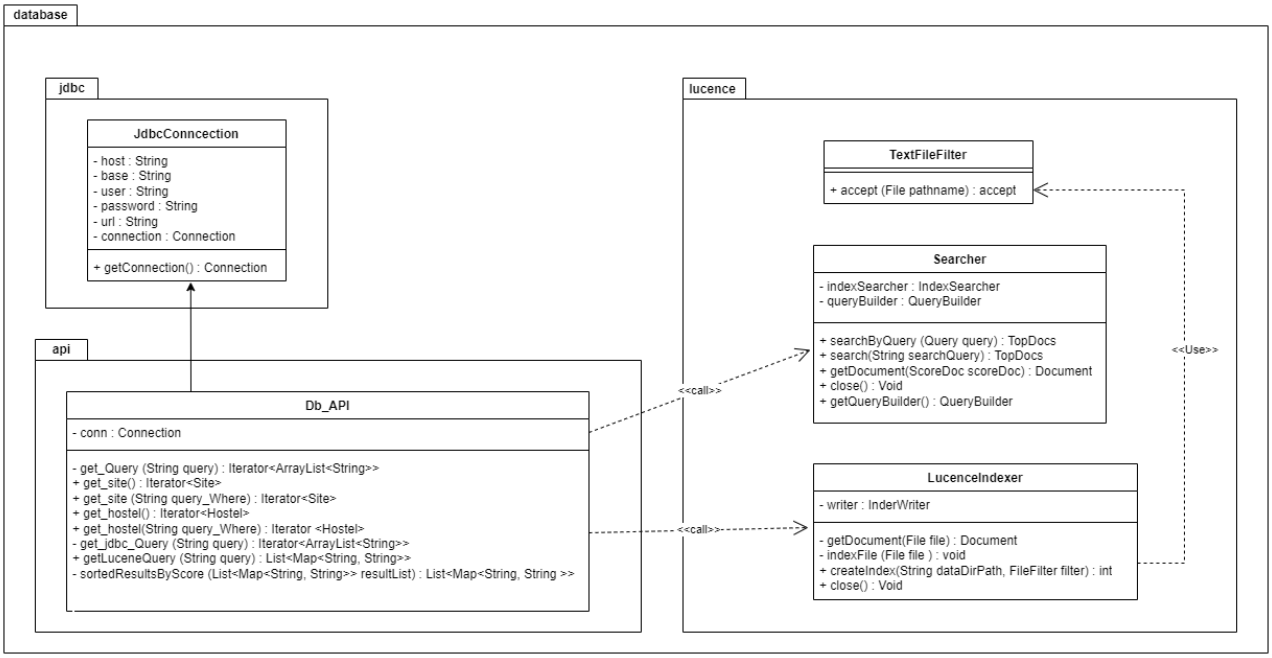

FIGURE 3 – Diagramme UML de la partie Database

L'API est structurée en trois sous-packages distincts pour assurer des fonctionnalités spécifiques :

Package "lucene" : Ce package gère les fichiers indexés et s'occupe du traitement des requêtes liées à la recherche. Il est responsable de la gestion des opérations de recherche et de l'optimisation des résultats.

Package "jdbc" : Ce package se concentre sur le traitement des requêtes "normales" liées à la base de données MySQL. Il prend en charge les opérations courantes d'interaction avec la base de données, telles que l'insertion, la mise à jour, et la récupération de données.

Package "api" : Fonctionnant comme une façade pour la couche métier, ce package connecte les deux autres packages. Il agit comme un point d'entrée central pour les fonctionnalités métier en orchestrant les appels aux packages "lucene" et "jdbc". Ainsi, il offre une interface unifiée pour les opérations métier tout en tirant parti des fonctionnalités spécifiques de chaque sous-package.

3.1 Partie Lucene

Dans ce package, nous avons élaboré des classes responsables de l'indexation d'un répertoire. Ces classes offrent également la fonctionnalité d'ajouter des fichiers texte au répertoire ainsi qu'à l'index. De plus, elles permettent d'effectuer des recherches dans l'index en utilisant des mots-clés. En deuxieme méthode (createIndex) important création de l'index pour un répertoire de fichiers. Cette méthode parcourt tous les fichiers d'un répertoire donné, applique un filtre pour sélectionner certains fichiers

classe "LuceneIndexer" Il y a une conversion d'un fichier en un document Lucene. Une des méthode crée un document Lucene en lisant le contenu d'un fichier donné. Elle ajoute plusieurs champs au document : le contenu du fichier, le nom du fichier, et le chemin du fichier. Le contenu est indexé pour permettre la recherche, tandis que le nom et le chemin du fichier sont stockés pour référence future.

Les classes principale de la partie Lucene Pour assurer cette partie, on a trois classes principales :

- TextFileFilter : Création d'un filtre de fichiers pour avoir uniquement les fichiers avec l'extension (.txt)
- LuceneIndexer : Création d'un index Lucene à partir des fichiers d'un répertoire Description avec 'Index Writer'
- Searcher : cette classe sert à :
  - Recherche dans l'indexe Lucene offrant une interface pour créer des requetes
  - Exécution les recherche
  - Obtention les résultats
  - Accès aux documents et gérer la fermeture de l'index

3.2 Partie JDBC

La classe JdbcConnection du package "jdbc" agit en tant que gestionnaire de connexion à la base de données MySQL. Elle encapsule les paramètres de connexion tels que l'adresse du serveur, le nom de la base de données, l'utilisateur, et le mot de passe. Utilisant le modèle singleton, elle garantit l'existence d'une unique instance de connexion dans l'application. Cette classe offre ainsi une approche centralisée et efficace pour gérer les connexions à la base de données dans le cadre du package "jdbc".

3.3 Partie API

La classe DbAPI sert d'interface pour interagir avec une base de données et un index Lucene. Elle offre des méthodes pour exécuter des requêtes SQL via JDBC et des requêtes textuelles via Lucene, et pour récupérer des données structurées (sites touristiques et hôtels) en les transformant en objets Java. Elle gère également la fusion des résultats de recherche JDBC et Lucene, permettant ainsi des requêtes hybrides et une récupération de données enrichie.

La classe Db_API dans ce code joue le rôle de façade entre la partie SQL (JDBC) et la partie Lucene. La composition entre les deux parties se fait principalement dans la méthode getLuceneQuery.

- **Extraction de la requête SQL et Lucene :** La méthode commence par diviser la requête initiale en deux parties distinctes : jdbc (représentant la requête SQL) et lucene (représentant la requête Lucene) en utilisant le mot-clé "WITH" comme séparateur.
- **Exécution de la requête SQL :** La partie SQL est traitée en appelant la méthode privée getQuery(jdbc), qui renvoie un itérateur sur les résultats de la requête SQL. Ces résultats sont stockés dans itjdbc.
- **Exécution de la requête Lucene :**La partie Lucene est traitée en utilisant la classe Searcher pour effectuer une recherche dans l'index Lucene en fonction de la requête Lucene spécifiée. Les résultats sont stockés dans topDocs.
- **Combinaison des résultats SQL et Lucene :**Les résultats des deux sources sont combinés en comparant les clés de correspondance (par exemple, le nom de fichier) entre les résultats JDBC (itjdbc) et Lucene (setlucenetoIT).Pour chaque paire correspondante, une nouvelle Map est créée, contenant des informations provenant des deux sources.
- **Tri des résultats par score Lucene :**Les résultats finaux sont triés en fonction du score Lucene, avec les résultats les mieux notés apparaissant en premier.
- **Renvoi des résultats combinés et triés :**La méthode retourne la liste finale des résultats, où chaque élément est une Map contenant les informations fusionnées des sources SQL et Lucene, triée par ordre décroissant de score Lucene.

4 Plans d'exécution

4.1 Premier plan d'exécution

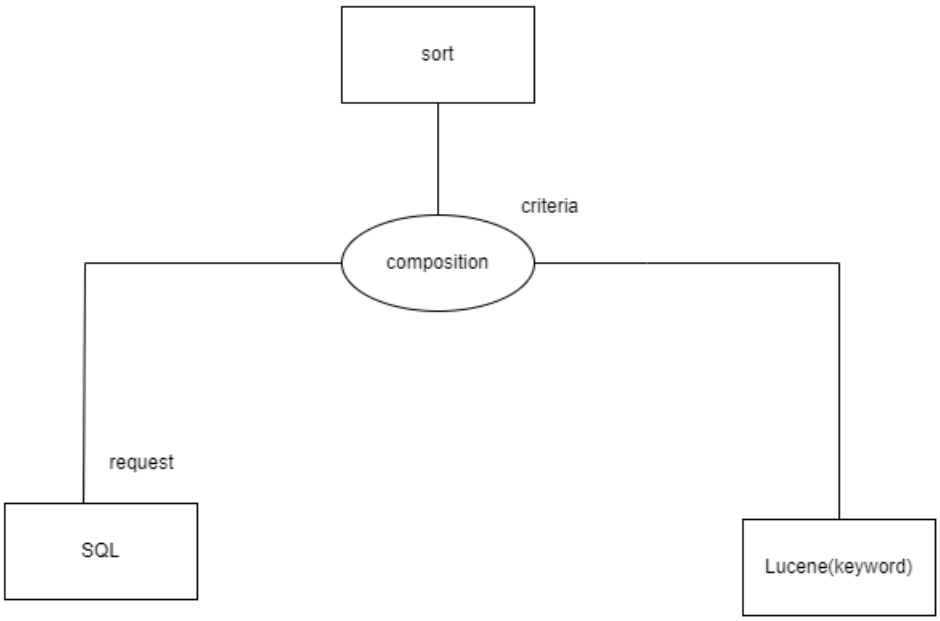

FIGURE 4 – Plan d'exécution

Pour cette première étape d'exécution, nous traitons séparément la partie SQL de la requête et la partie textuelle. Ensuite, nous fusionnons les résultats qui correspondent en termes de valeurs lors de la jointure. Étant donné que les résultats de la requête textuelle sont déjà triés en ordre décroissant par Lucene, il est préférable de commencer par parcourir les résultats de l'index textuel, puis de les comparer avec les résultats de la requête SQL. Si une correspondance est trouvée, nous effectuons la jointure.

4.2 Deuxième plan d'exécution

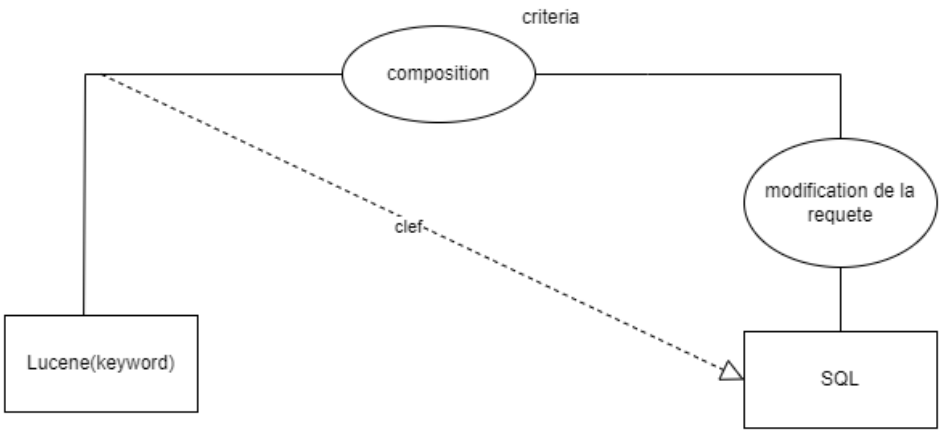

FIGURE 5 – Plan d'exécution

Pour la deuxième partie, nous commençons par exécuter la requête textuelle. Pour chaque clé renvoyée par cette requête, nous effectuons ensuite une requête SQL pour vérifier si elle est incluse dans le résultat final ou non.