RAPPORT DE PROJET BASES DE DONEES AVANCEES

Année scolaire : 2018/2019

Recherches des informations de voyages

Membres du groupe :

BAYO Mohamed II
FAGOT Benoit
GUEDJOU Yasmine
SAMAH Wehab
TENSAOUT Thiziri

Encadré par :

LIU Tianxiao
VODISLAV Dan

Table des matières

I. Introduction

Dans le cadre de l'atelier de gestion de projet pour la partie bases de données avancées nous avons comme objectif de construire une extension d'une BD relationnelle, qui, en plus de répondre à des requêtes SQL, puisse aussi traiter des requêtes portant sur du contenu textuel. Cette extension doit s'appuie sur la BD relationnelle MySQL et sur le moteur d'indexation textuelle Lucene.

II. Présentation de la base de données

Pour la réalisation de la base de données nous avons choisi d'utiliser le système de gestion de base de données MySQL.

- **Modèle conceptuel de données (MCD)**

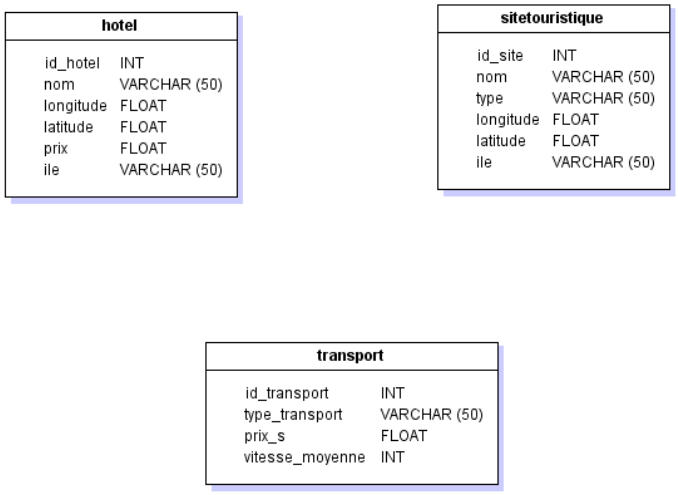

Figure 1 : Modèle conceptuel de données

- **Modèle logique de données (MLD)**

Sitetourisitique (<u>id_site</u>, nom, type, longitude, latitude, ile)
Hôtel (<u>id_hotel</u>, nom, type, prix, longitude, latitude, ile)
Transport (<u>id_transport</u>, type, prix_transport, vitesse)

- **Dictionnaire de données**

Num	Nom	Code	type	taille
1	id_hotel	ID_HOTEL	Int	
2	nom	NOM	Varchar	50
3	longitude	LONGITUDE	Float	
4	latitude	LATITUDE	Float	
5	prix	PRIX	Float	
6	ile	ILE	Varchar	50
7	id_site	ID_SITE	Int	
8	type	TYPE	Varchar	50
9	id_transport	ID_TRANSPORT	Int	
10	type_transport	TYPE_TRANSPORT	Varchar	50
11	prix_s	PRIX_S	Float	
12	vitesse_moyenne	VITESSE_MOYENNE	Int	

Figure 2 : Dictionnaire des données

III. API de la BDe

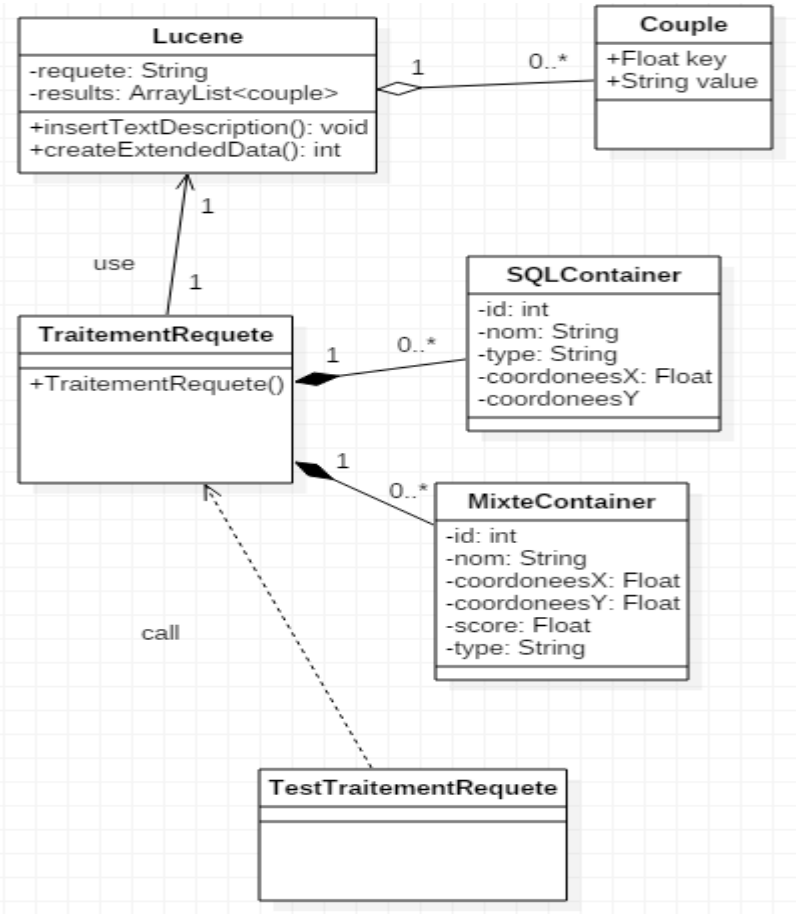

Figure 3 : diagramme de classe UML représentant le fonctionnement de l'API de la Bde

Fonctionnement API

1) Création des descriptions automatisé des fichiers contenant les descriptions textuelles des sites touristiques, sous la forme « Key ».Txt, par la classe Lucene.java, stockés dans le répertoire R

2) Indexation des fichiers, indexes stockés dans le répertoire Index (toujours Lucene.java)

3) Traitement de la requête (TraitementRequete.java)

→ Si requête mixte : rajouter la « Key » dans la requête :

→ l'objet Lucene de « TraitementRequete.java » reçoit la requête textuelle (ce qu'il y a après le WITH)

→ Exécution de la requête textuelle

→ stocker les résultats (key, score) dans une liste d'objets Couple.java

→ Exécution de la requête SQL

→ Pour chaque ligne de la requête SQL, chercher dans la List<Couple> un couple qui vérifie Site.key = Couple.key

→ Si la ligne vérifie cette jointure, alors on stocke la projection de cette jointure dans une liste d'objets MIXTEConainer.java

→ Si requête SQL simple :

→ stocker chaque ligne dans une liste d'objets SQLContainer.java

IV. Résultats d'une requête mixte

Requête avec la clause where :

```
SELECT id_site, nom, type, longitude, latitude
FROM SiteTouristique WHERE type = \\ 'historique\\'
WITH ile~2
```

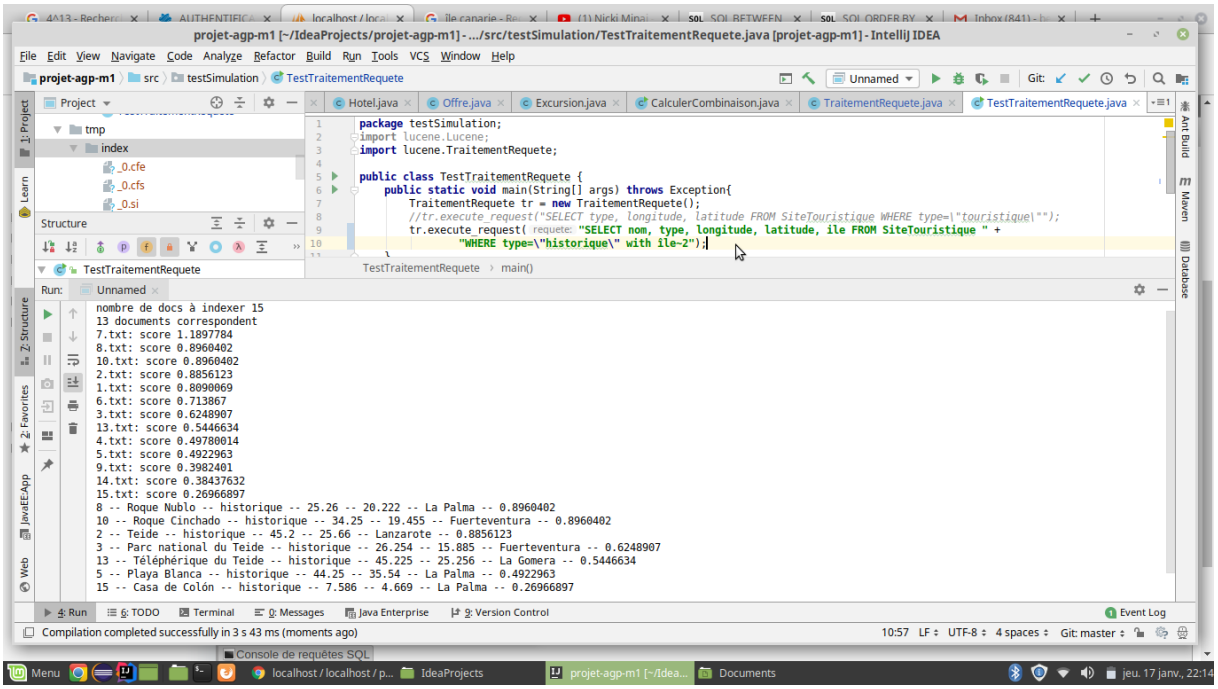

Figure 4 : capture d'écran du résultat de la requête mixte

V. Etude comparative entre les deux plans

Pour l'étude comparative entre les deux plans proposés dans le cahier de charge nous avons étudié le plan d'exécution de la requête mixte suivante :

```
SELECT id_site, nom, type, longitude, latitude
FROM SiteTouristique
WITH ile~2
```

a) Plan d'exécution du plan 1

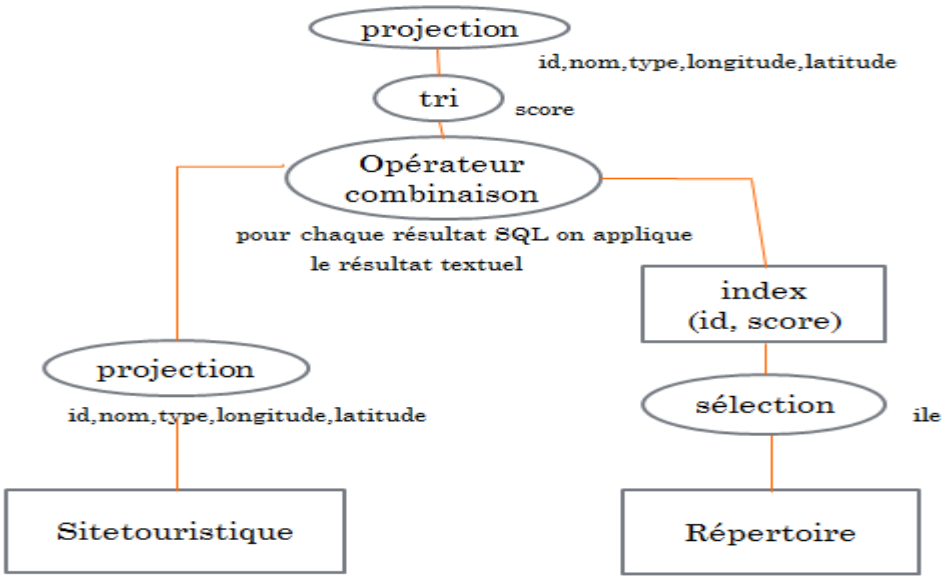

Figure 5 : Plan d'exécution du premier plan

b) Plan d'exécution du plan 2

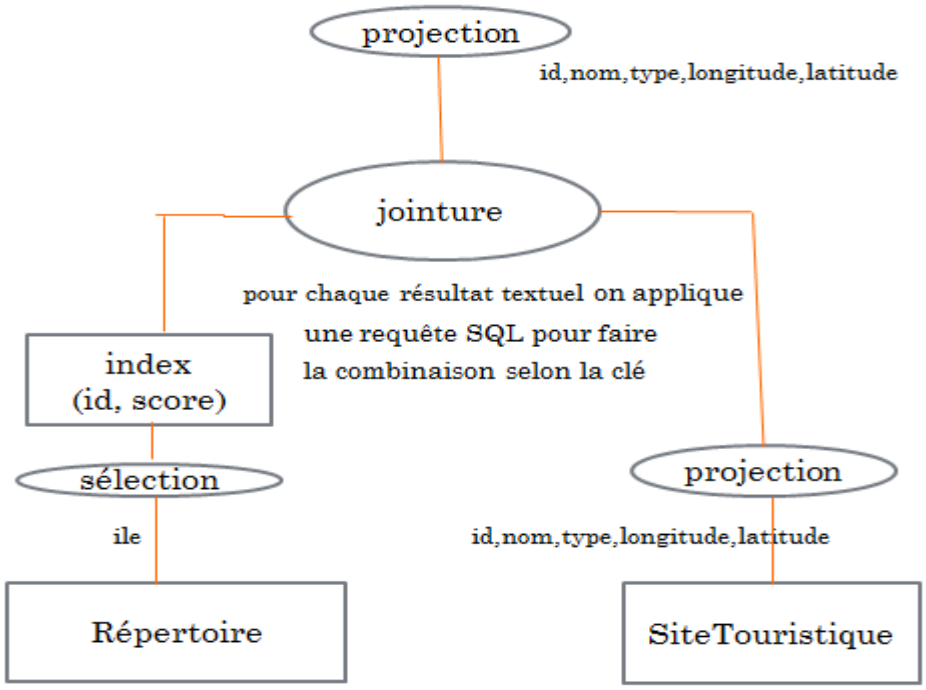

Figure 6 : Plan d'exécution du deuxième plan

c) Comparaison entre les deux plans

	Plan 1	**Plan 2**
Nombre de lignes	13	**13**
Temps de calcul	**Texec=t1+t2+n*m*Tseq+cout_tri**	**Texec=t1+t2+n*TreqSQL**
Résultat	**Texec= 46+266+3+1=316ms**	**Texec=46+266+13*266=3770ms**

Tableau représentant la comparaison entre les deux plans

On remarque le premier plan est beaucoup plus optimal que le deuxième plan en terme de temps d'exécution.

VI. Amélioration et optimisation

Pour l'optimisation des requêtes on pourrait ajouter un index sur la table SiteTouristique sur l'id de la table ce qui nous permettrais de réduire le temps de recherche de le requête SQL et par conséquent réduire le temps d'exécution.