

Rapport TME JDBC

PAR : Hanane DJEDDAL - 3803192

Introduction :

Le but de ce TME est d'implémenter des différents algorithmes de jointure sous JDBC en utilisant Le SGBD H2 et L'interface graphique SQLWorkbench.

Exercice 1:

On utilise une base de données contenant deux tables : Ens, qui représente les enseignants, un enseignant a un identifiant, un nom et un prénom. Et la table Module, un module a un identifiant, un nom et l'enseignant responsable du module. Un enseignant peut être le responsable de plusieurs modules.

Ci-dessous, les commandes SQL pour créer les tables sur SQLWorkbench :

```
CREATE TABLE Ens
(id_ens INTEGER PRIMARY KEY AUTO_INCREMENT,
 nom VARCHAR (255),
 prenom VARCHAR (255));
```

```
CREATE TABLE Module
(id_module INTEGER PRIMARY KEY AUTO_INCREMENT,
 nom_module VARCHAR (255),
 ens INTEGER,
 FOREIGN KEY(ens) REFERENCES Ens(id_ens));
```

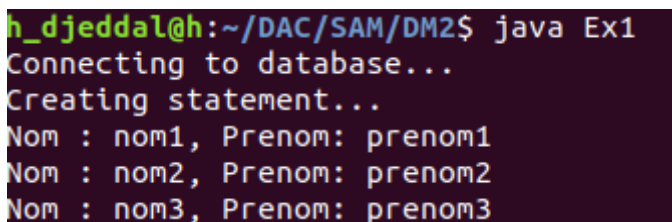
```
INSERT INTO Ens (nom,prenom) VALUES ('nom1','prenom1');
commit;
```

...

```
INSERT INTO Module (nom_module,ens) VALUES ('mod1',1);
commit;
```

...

Le code Ex1.java permet d'afficher la table Ens (nom, prenom) avec JDBC.



```
h_djeddal@h:~/DAC/SAM/DM2$ java Ex1
Connecting to database...
Creating statement...
Nom : nom1, Prenom: prenom1
Nom : nom2, Prenom: prenom2
Nom : nom3, Prenom: prenom3
```

Exercice 2:

Dans cet exercice, on implémente la jointure par boucles imbriquées. Pour cela:

- On définit une requête paramétrée : "SELECT * FROM Module WHERE ens =? "
- On parcourt la table Ens séquentiellement, et pour chaque itération :
 - On récupère id_ens.
 - On exécute la requête paramétrée, en passant id_ens comme argument.

```
h_djeddal@h:~/DAC/SAM/DM2$ java Ex2
Connecting to database...
Creating statement...
ID_ENS  NOM    PRENOM  ID_MODULE  NOM_MODULE
1       nom1   prenom1  1          mod1
1       nom1   prenom1  4          mod4
2       nom2   prenom2  2          mod2
2       nom2   prenom2  5          mod5
3       nom3   prenom3  3          mod3
Jointure effectuée en: 0.074 secondes.
```

Le temps d'exécution est 0.074seconds

Exercice 3:

Dans cet exercice, on implémente la jointure par tri-fusion sur les deux tables en créant chaque table dans une base (2 site différents), pour cela :

Sur SQLWorkbench : On cree deux bases de données avec les URL :

jdbc:h2:tcp://localhost:9093/~/base1

jdbc:h2:tcp://localhost:9094/~base2

En Java :

- On cree une connexion pour chaque base de données.
- On recupère chaque table de sa base de données.
- On trie les deux tables.
- On applique l'algorithme de tri-fusion.

```
h_djeddal@h:~/DAC/SAM/DM2$ java Ex3
Connecting to database...
Creating statement...
ID_ENS  NOM    PRENOM  ID_MODULE  NOM_MODULE
1       nom1   prenom1  1          mod1
1       nom1   prenom1  4          mod4
2       nom2   prenom2  2          mod2
2       nom2   prenom2  5          mod5
3       nom3   prenom3  3          mod3
Jointure effectuée en: 0.054 secondes.
```

On remarque que le temps d'exécution avec le tri-fusion (0.054 secondes) est inferieur à celui de la jointure imbriquée.