Mini projet 2023 “Jointure avec index sur disque”

4ASTI – INSA CVL – B. Nguyen & P. Clemente

Code source

Un zip du code est disponible sur :
https://www.benjamin-nguyen.fr/ENS/4ASTI-BD2/projetSGBD/miniprojet.zip

On dispose d’une classe Tuple.

On dispose de classes implémentant l’interface Operateur :

- TableSimple (opérateur FullScan en mémoire)

- TableDisque (opérateur FullScan sur disque)

- FiltreEgalite (opérateur restriction/filtre attribut = valeur)

- DBI (opérateur de jointure double boucle imbriquée sans utilisation d’index)

On disposé d’un programme main exemple : ExempleTableDisque.java

On dispose de deux fichiers (table1 et table2) qui représentent les données sur le disque, à mettre au bon endroit de votre arborescence de fichiers et à changer dans le programme ExempleTableDisque Vous pouvez comprendre le format de ces fichiers en regardant FiltreDisque et en regardant le contenu des fichiers en format hexadecimal (par exemple utilisation de la commande format-hex sous windows ou hexdump sous linux)

Objectifs généraux du projet

L’objectif du projet est en mettre en application vos connaissances sur les mécanismes internes des SGBD par l’implémentation d’un opérateur de jointure utilisant un index sur le disque. En particulier, lors des TDs précédents nous avons utilisé des tables en mémoire, nous allons désormais utiliser des tables sur disque.

Question 1 : Création de l’index

Implémentez un index à base de table de hachage, qui peut être construit sur un fichier déjà existant (par exemple table1) qui indique dans quelle page du disque se trouve les enregistrements dont on connaît la valeur de l’attribut clé. La position d’un enregistrement est connue sur le disque et on peut se positionner au bon endroit en utilisant la classe RandomAccessFile pour ouvrir le fichier. Donnez un fichier exemple montrant la bonne création de l’index.

Question 2 : Utilisation de l’index

Créez un opérateur qui va combiner l’appel à l’index et direct access qui prend en entrée une position dans le fichier, l’attribut clé et la valeur, et qui charge la page en mémoire en utilisant un accès direct puis retourne le tuple. Note : essayez de prendre en compte le fait que la page peut déjà être en mémoire. Les paramètres liés au nombre de blocs, leur taille etc sont stockés dans la classe

TableDisque. Donnez un fichier exemple et calculez (automatiquement, en vous inspirant du code dans TableDisque) combien de blocs sont lus sur le disque.

Question 3 : Jointure sur index

Implémentez une jointure de type boucle sur index et donnez un fichier exemple.

Question 4 : Pour aller plus loin

Implémentez un index B+ et proposez un fichier démontrant son fonctionnement, en indiquant notamment le nombre de pages lues sur le disque. L'index sera stocké intégralement en mémoire.

Question 5 : Pour aller encore plus loin

Améliorez votre index B+ en ne conservant en mémoire que le bloc racine de l'arbre. Proposez un programme de démonstration qui indique le nombre de blocs lus sur le disque (blocs de l'index et de données)