RAPPORT TP 1 & 2 (SAM)

29 janvier 2024

Hichem BOUZOURINE (21319982)

Table des matières

1 Accès aux données avec indexes

Le TP 1 et 2 ont été axés sur l'accès aux données en utilisant des index, avec un accent particulier sur l'organisation des données en pages et la manipulation d'index pour optimiser les opérations de recherche et de mise à jour. Dans ce rapport, nous allons explorer les différentes étapes algorithmiques de chaque exercice, expliquer les choix effectués et détailler le fonctionnement des principales fonctions développées.

1.1 Créer un index

L'objectif de cet exercice était de créer des index pour différents attributs de la table. Deux types d'index ont été considérés : un index **unique** pour l'attribut *a0* et un index **non unique** pour l'attribut *a2*. Pour l'index unique, chaque entrée associe une clé (la valeur de l'attribut) à un rowid, tandis que pour l'index non unique, chaque clé peut être associée à plusieurs **rowids**.

Fonctions principales :

— **creation_index_unique** : Crée un index unique pour un attribut donné (*a0* dans notre cas d'exemple). Les clés sont les valeurs de l'attribut, et les valeurs sont les rowids associés.

— **creation_index** : Crée un index non unique pour un attribut donné (*a2* dans notre cas d'exemple). Les clés sont les valeurs de l'attribut, et les valeurs sont une liste de rowids associés.

1.2 Accès par index

Cet exercice a exploré différentes méthodes d'accès aux données en utilisant les index créés. Il s'agit principalement d'accès ciblés en fonction des valeurs des attributs indexés, en utilisant des méthodes de recherche séquentielle et des méthodes d'accès par intervalle.

Fonctions principales :

— **acces_par_index_unique** : Accède à un tuple en utilisant un index unique. La fonction prend en compte la recherche d'une valeur spécifique dans l'index et renvoie le tuple correspondant.

— **acces_par_index** : Accède à plusieurs tuples en utilisant un index non unique. La fonction prend en compte la recherche d'une valeur spécifique dans l'index et renvoie tous les tuples correspondants.

— **acces_intervalle_par_index_unique** : Accède à un intervalle de tuples en utilisant un

index unique. La fonction prend en compte les bornes d'un intervalle et renvoie tous les tuples dont la clé dans l'index est comprise dans cet intervalle.

— **acces_intervalle_par_index** : Accède à un intervalle de tuples en utilisant un index non unique. La fonction prend en compte les bornes d'un intervalle et renvoie tous les tuples dont la clé dans l'index est comprise dans cet intervalle.

1.3 Mise à jour de données

Cet exercice a abordé la mise à jour de données dans la table, avec la possibilité de modifier la valeur d'un attribut pour un ou plusieurs tuples. Les fonctions développées ont pris en compte la modification d'un seul tuple et la modification de plusieurs tuples basée sur une condition.

Fonctions principales :

— **update_unique** : Modifie la valeur d'un attribut pour un seul tuple. La fonction prend en compte la recherche d'une valeur spécifique dans un index unique, puis met à jour le tuple correspondant.

— **update_plusieurs** : Modifie la valeur d'un attribut pour plusieurs tuples basés sur une condition. La fonction prend en compte la recherche d'une valeur spécifique dans un index non unique, puis met à jour tous les tuples correspondants.

1.4 Persistence

Cet exercice s'est concentré sur la persistance des index, en stockant les entrées triées dans des pages pour faciliter la reconstruction rapide des index. Deux types d'index ont été considérés : un index unique et un index non unique.

Fonctions principales :

— **sauvegarder_index_unique** : Sauvegarde un index unique dans des pages. La fonction prend en compte la taille de la page et stocke les entrées triées de l'index.

— **sauvegarder_index** : Sauvegarde un index non unique dans des pages. La fonction prend en compte la taille de la page et stocke les entrées triées de l'index.

— **charger_index_unique** : Charge un index unique à partir des pages sauvegardées. La fonction lit les entrées triées depuis les pages et reconstruit l'index.

— **charger_index** : Charge un index non unique à partir des pages sauvegardées. La fonction lit les entrées triées depuis les pages et reconstruit l'index.

— **update_index_unique** : Mise à jour d'un index unique lorsqu'un attribut est modifié. La

fonction prend en compte la modification de l'attribut et actualise l'index correspondant.

1.5 Index bitmap

Cet exercice a introduit le concept d'index bitmap pour deux attributs (a5 et a6). Les index bitmap sont représentés sous forme matricielle, où chaque ligne correspond à une valeur de l'attribut et chaque colonne correspond à un tuple. Un bit à 1 indique que la valeur de l'attribut correspondant est présente pour le tuple, et un bit à 0 indique l'absence.

Fonctions principales :

— **creation_index_bitmap** : Crée un index bitmap pour un attribut donné. La fonction parcourt tous les tuples, identifie les valeurs distinctes de l'attribut et construit la matrice bitmap.
— **acces_par_index_bitmap** : Accède à des tuples en utilisant un index bitmap. La fonction prend en compte la recherche d'une valeur spécifique dans l'index bitmap et renvoie les tuples correspondants.
— **acces_intervalle_par_index_bitmap** : Accède à un intervalle de tuples en utilisant un index bitmap. La fonction prend en compte les bornes d'un intervalle et renvoie les tuples correspondants.

1.6 Conclusion

Les TPs 1 et 2 ont permis d'explorer en profondeur les concepts d'indexation et d'optimisation des opérations sur les bases de données. La mise en œuvre de différents types d'index, la gestion des accès aux données par index, la mise à jour efficace des données et la persistance des index ont été couvertes de manière approfondie. En outre, l'introduction des index bitmap a élargi notre compréhension des structures d'index non conventionnelles. Ces compétences sont cruciales pour concevoir des systèmes de gestion de bases de données performants et évolutifs. Ce travail pratique a fourni une expérience pratique et approfondie, renforçant ainsi notre compréhension des concepts fondamentaux en matière de gestion de bases de données relationnelles.