TP: Indexation

Pierre Senellart (pierre.senellart@ens.fr)

10 mai 2017

Le but de ce TP est de se familiariser avec les structures d'index classiques dans les systèmes de gestion de bases de données.

Rendu

Le rendu de ce TP doit être fait par mél, avant 23:59 le vendredi 12 mai, à pierre.senellart@ens.fr en fournissant, en attachement du mél, un fichier texte ou PDF contenant un compte-rendu de vos réponses aux questions. Deux jours supplémentaires sont accordés par rapport à d'habitude en raison d'un conflit d'emploi du temps affectant certains étudiants. Des pénalités seront comptées en cas de rendu tardif (0,5 point par heure de retard).

1 Arbres B+

1a. Construire un arbre B+ pour l'ensemble suivant de valeurs :

$$2, 3, 5, 7, 11, 17, 19, 23, 29, 31$$

On suppose que l'arbre est initialement vide, et que ces valeurs sont insérées par ordre croissant. On construira un tel arbre B+ pour chacun des nombres suivants de pointeurs par nœud :

1aα) 4

1aβ) 6

1aγ) 8

1b. Pour chacun des trois arbres B+ obtenus, construire l'arbre B+ obtenu après la succession des opérations suivantes :

— Insérer 9

— Insérer 10

— Insérer 8

— Supprimer 23

— Supprimer 19

2 Hachage extensible

2a. Construire une table de hachage extensible pour l'ensemble suivant de valeurs :

$$2, 3, 5, 7, 11, 17, 19, 23, 29, 31$$

On suppose que la fonction de hachage est $h : x \mapsto x \bmod 8$ et que chaque alvéole (*bucket*) peut contenir trois valeurs.

2b. Construire la table de hachage extensible obtenue à partir de celle de la question précédente après la succession des opérations suivantes :

- Supprimer 11
- Supprimer 31
- Insérer 1
- Insérer 15

3 Index en PostgreSQL

En PostgreSQL, on peut choisir le type d'index utilisé dans un `CREATE INDEX` avec `USING method` entre le nom de la table et la liste des colonnes, `method` pouvant être `btree` (arbre B, une variante des arbres B+ dans laquelle les données peuvent être stockées dans les nœuds internes) et `hash` (une table de hachage).

3a. Sur la table `unicode` du TP précédent, comparer la performance des deux types d'index pour des requêtes variées. Également tenir compte du temps de création de l'index. Commenter.

3b. Il existe trois manières classiques de faire croître une table de hachage au fur et à mesure que des éléments y sont insérés :

- Approche naïve : doubler la taille de la taille en la recréant intégralement, quand nécessaire
- Hachage extensible : approche vue en cours
- Hachage linéaire : `https://en.wikipedia.org/wiki/Linear_hashing`

En examinant le code source de PostgreSQL, consultable depuis `https://doxygen.postgresql.org/` ou téléchargeable depuis `https://www.postgresql.org/ftp/source/`, déterminer quelle méthode est utilisée. Justifier la manière dont vous avez déterminé ceci.