Architectures des Systèmes de Bases de Données
TP5 Hash Join

Qian Christian | christian-qian@live.fr | Etudiant : 21964319

Le projet est composé de cinq fichiers plus le fichier de test :

- l'interface HashProbing devant être implémenté par les classes de Hashage.

- la classe HashLinearProbing parcourant les buckets de un à un lors d'une collision ou du rehashage.

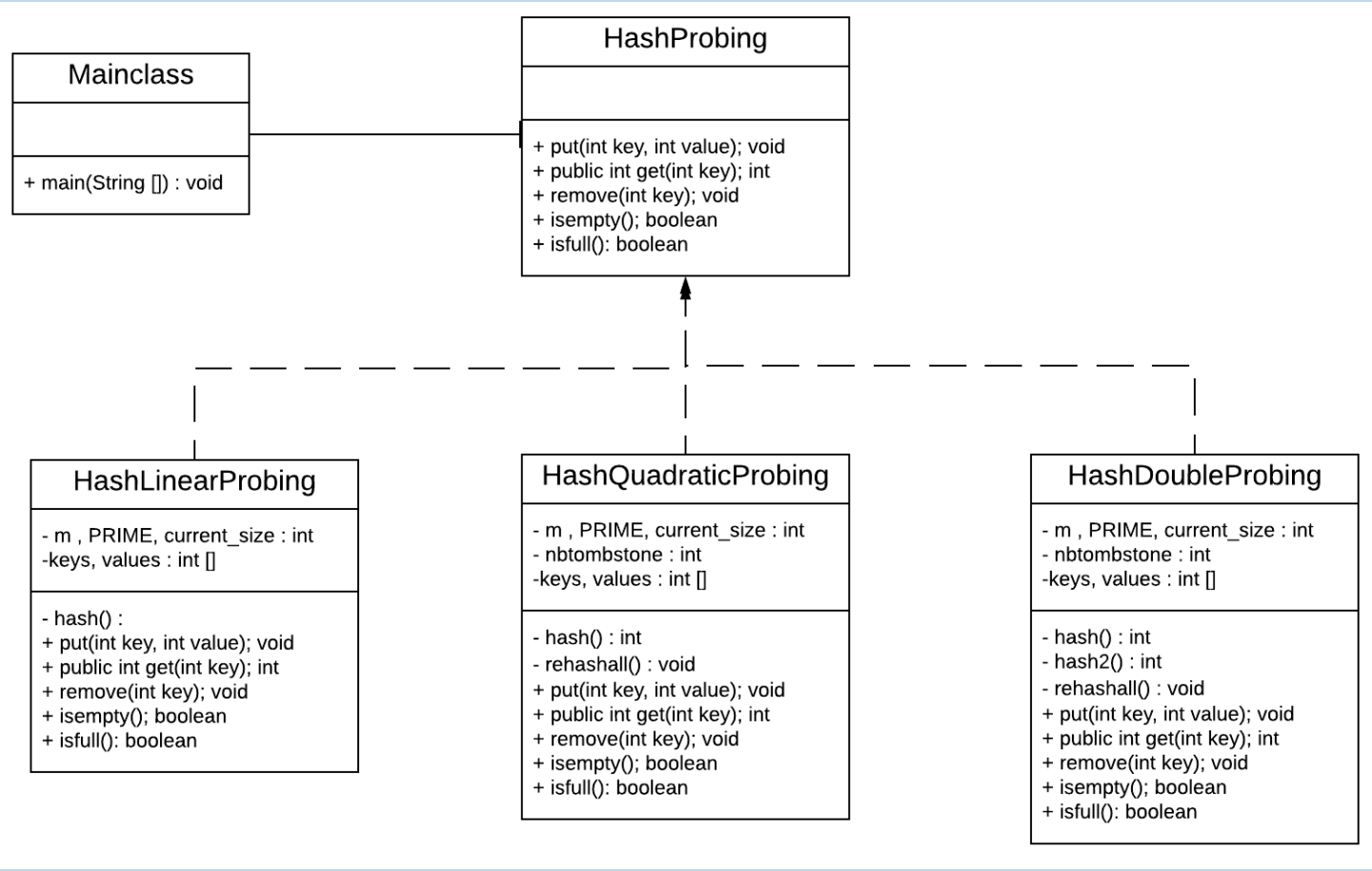
- la classe HashQuadraticProbing parcourant les buckets en incrémentant l'indice i de un et en prenant son carré. Lorsque le tableau a un grand nombre de valeur, le probing path peut retomber sur les même buckets sans trouver d'emplacements vides. Pour éviter alors les boucles infinies , on ajoute la technique du "alternating signs". Il s'agit alterner le signe de i^2 a chaque incrémentation, permettant ainsi de couvrir chaque bucket. Elle est valide seulement si la taille du tableau m est congruent à 3 modulo 4 (3, 7, 11, 19, ...).

- la classe HashDoubleProbing parcourant les buckets en incrémentant l'indice i de un et avec un facteur d'une deuxième fonction de hashage. De la même manière que pour HashQuadraticProbing, il faut éviter les boucles infinies. Cette deuxième fonction prend la formule **hash2(key) = PRIME2 – (key % PRIME2)**, la valeur de PRIME2 est un nombre premier tel que, 2 < PRIME2 < PRIME, avec PRIME étant le nombre premier de la première fonction de hashage.

- la classe Mainclass créant des objects de type HashProbing et de sous-type HashLinearProbing, HashQuadraticProbing ou HashDoubleProbing. Elle contient une fonction main, pour tester l'ajout de paires clé-indice, le get des indices à partir des clés, le get et le remove de clés absentes.

Pour les classes HashQuadraticProbing et HashQuadraticProbing, le principe de tombstone est ajouté pour éviter qu'un probing path devienne inutilisable lorsqu'un élément est retiré. En remplacant l'élément retiré par un tombstone, le bucket est compté comme plein pendant le get et remove, mais vide pendant un put.
Cependant, trop de tombstone augmente le temps de recherche, lorsque le nombre de tombstone dépasse la mointié du tableau, on les supprime en effectuant un rehashage complet de la table.

Diagramme UML



Le Test

Pour effectuer le Hash join, on ajoute les valeurs de R dans les objets avec les trois types de Hashage. Puis on créer un tableau result avec le signe attendu de la valeur retourné par la fonction get. En parcourant le tableau S, on compare le signe de get, si la clé est absente on a -1 donc < 0 , et >= 0 si elle est présente.
En effectuant get sur l'ensemble de S, on obtient bien un pass pour tous les types de hashage.