Licence et Master d'Informatique – S6/M1 – 2006/2007

Bases de données avancées

juin 2007

Examen – session 2

Durée : 3h, documents et calculatrice autorisés

Exercice 1 : (**Indexation**) Considérez l'index B+ de la figure 1, dont les entrées de données suivent l'alternative 1 et les trois premiers sous-arbres (dénomés A, B et C) sont non spécifiés. Chaque noeud interne peut contenir un maximum de 4 valeurs et 5 pointeurs. Chaque feuille peut contenir jusqu'à 4 entrées de données, et les feuilles sont doublement chaînées (ce double chaînage n'est pas représenté sur la figure). Répondez, en **justifiant**, aux questions suivantes (toutes les questions sont indépendantes et font référence à l'index de la figure 1) :

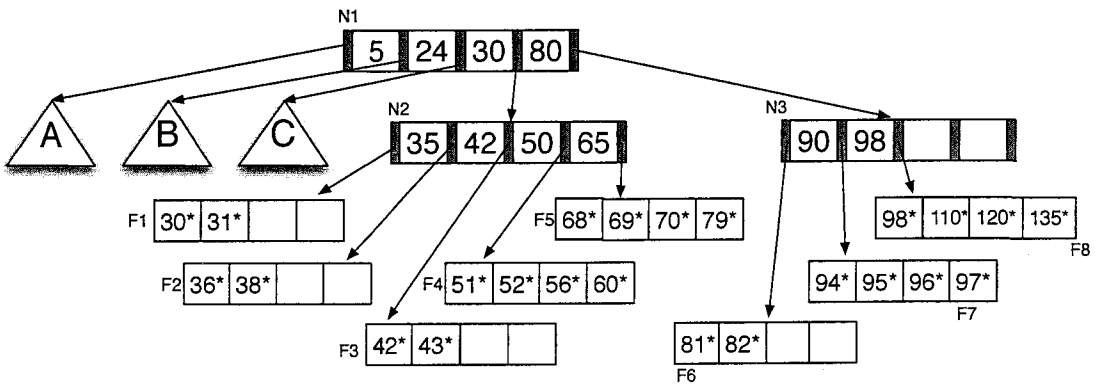

FIG. 1 – Index B+

Question 1.1 : Donnez le nom des noeuds (internes et feuilles) qui doivent être rappatriés du disque pour répondre à la requête : " Obtenir tous les enregistrements avec une valeur de clé de recherche supérieure à 38 ".

Question 1.2 : Donnez l'index B+ qui résulterait de l'insertion d'un enregistrement avec 100 pour valeur de clé de recherche.

Question 1.3 : Donnez l'index B+ qui résulterait de la suppression d'un enregistrement avec 81 pour valeur de clé de recherche.

Question 1.4 : Donnez une valeur de clé de recherche telle que son insertion dans l'index de la figure 1 provoquerait une incrémentation de la hauteur de l'index.

Question 1.5 : Bien que les sous-arbres A, B et C ne soient pas spécifiés, que pouvez-vous dire du contenu et de la forme de ces sous-arbres ?

Exercice 2 : (**Indexation**) On considère la relation suivante :

`etudiant(num:integer, nom:string, idf:integer)`

et on suppose qu'on dispose d'un index clusterisé sur `num` et d'un index non-clusterisé sur `nom`. Répondez aux questions suivantes en **justifiant** vos réponses.

Question 2.1 : Donnez un exemple de mise-à-jour clairement accélérée grâce aux (ou à un) index.

Question 2.2 : Donnez un exemple de mise-à-jour clairement ralentie à cause des (ou d'un) index.

Question 2.3 : Donnez un exemple de mise-à-jour ni accélérée, ni ralentie par les index.

Exercice 3 : (Optimisation de requêtes) On considère les relations suivantes :

- `etudiant(num:integer,nom:string,idf:integer)`
- `formation(idf:integer,nom:string,ufr:string,responsable:string)`
- `batiment(nom:string,ufr:string,os:string)`

où les clés primaires sont soulignées. L'attribut `idf` est un identifiant de formation (par exemple, l'identifiant 47 pourrait désigner le *master Sciences et Technologies, Mention Informatique, Spécialité Informatique*). L'attribut `ufr` désigne une unité de formation. La table `batiment` décrit les bâtiments dans lesquels les ufr dispensent des cours et le système d'exploitation (attribut `os`) qui équipe les salles de tp d'informatique gérées par une ufr dans un bâtiment.

Question 3.1 : Telles que les relations sont définies, un seul système d'exploitation peut équiper tous les ordinateurs d'une salle de tp gérée par une ufr dans un bâtiment donné. Pourquoi ?

Question 3.2 : Écrire une requête SQL donnant pour chaque étudiant le (ou les) bâtiment(s) équipé(s) de linux (s'il en existe) auxquels ils ont accès grâce à leur formation. Par exemple, `(Knuth, M5)` sera un enregistrement renvoyé par cette requête si l'étudiant Knuth est inscrit dans une formation en informatique (et en supposant que seul linux équipe les ordinateurs du M5...).

Question 3.3 : Exprimer cette requête dans l'algèbre relationel.

Question 3.4 : En supposant qu'il n'y a pas d'index, et que la jointure par boucles imbriquées simple est le seul algorithme de jointure disponible, donnez le plan d'exécution de cette requête qui vous semble être le meilleur. **Justifiez** votre réponse.

Question 3.5 : Supposons que :

- la taille d'un attribut est 10 octets ;
- la taille d'un bloc est 1000 octets ;
- il y a 10 000 étudiants ;
- il y a 55 formations ;
- il y a 100 bâtiments ;
- deux tiers des bâtiments sont équipés du système linux.

En précisant toute hypothèse supplémentaire que vous jugeriez être nécessaire, donnez une estimation du coût, en termes d'entrées/sorties, de votre plan d'exécution.

Exercice 4 : (Gestion des transactions) On considère deux séquences d'actions S1 et S2, listées dans l'ordre dans lequel elles sont soumises au SGBD :

```
S1=T1:W(B), T2:R(B), T2:R(A), T3:W(A), T1:W(A), T3:W(A),
   T1:commit, T2:commmit, T3:commit
S2=T1:W(B), T2:R(A), T2:R(B), T3:W(A), T1:W(A), T3:W(A),
   T1:commit, T2:commmit, T3:commit
```

Pour chacune de ces séquences et pour chaque type de contrôle de la concurrence donné ci-dessous, décrivez comment le gestionnaire de transactions traite ces actions. On suppose que les mécanismes d'estampillage attribuent la plus haute priorité à la transaction T1 puis à la transaction T2 en enfin à la transaction T3. Pour le contrôle par verrouillage, indiquez les requêtes de verrous exclusifs et partagés. Le gestionnaire de transactions traite les actions dans l'ordre donné. Si une transaction est bloquée, on suppose que toutes les actions de cette transaction sont mises dans une file d'attente jusqu'à ce que la transaction soit débloquée ; le gestionnaire de transactions continue alors avec l'action suivante (pour l'ordre de la séquence) d'une transaction non bloquée.

Question 4.1 : 2PL strict avec prévention des deadlocks par estampillage *Wait-die.*

Question 4.2 : 2PL strict avec prévention des deadlocks par estampillage *Wound-wait.*

Question 4.3 : 2PL strict avec détection des deadlock (donner, le cas échéant le graphe d'attente).

400001
400099

Licence et Master d'Informatique – S6/M1 – 2006/2007

Bases de données avancées

mai 2007

Examen – session 1

Exercice 1 : (Hachage extensible) On considère l'index par hachage extensible de la figure 1.
Question 1.1 : Donner l'index résultant de l'insertion (dans l'index de la figure) de la valeur hachée 17.
Question 1.2 : Donner l'index résultant de l'insertion (dans l'index de la figure) des valeurs hachées 11 et 15.
Question 1.3 : Donner l'index résultant de l'insertion (dans l'index de la figure) des valeurs hachées 8 et 24.
Question 1.4 : Donner l'index résultant de la suppression (dans l'index de la figure) de la valeur hachée 10.
Question 1.5 : Donner l'index résultant de la suppression (dans l'index de la figure) des valeurs hachées 28 et 12.

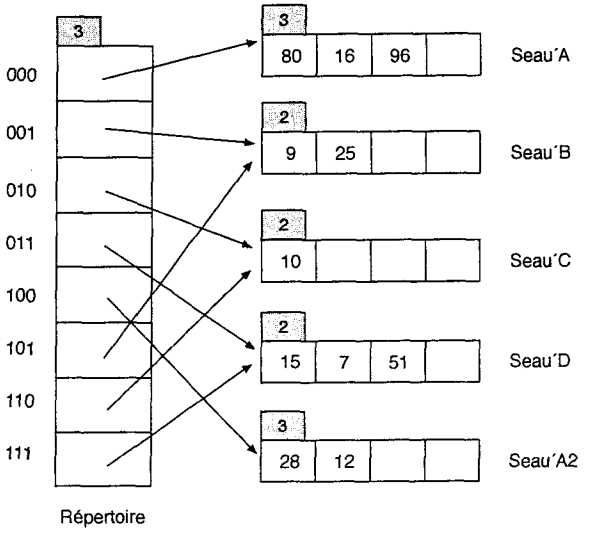

FIG. 1 – Index haché extensible

Exercice 2 : (Hachage linéaire) On considère l'index haché de la figure 2.
Question 2.1 : Quel est le nombre maximum d'entrées de données que l'on peut insérer avant qu'il y ait division d'un seau ?
Question 2.2 : Partant de l'index de la figure, donner une séquence minimal de valeurs dont l'insertion dans l'index provoque la division d'un seau. Représenter l'index à l'issue de ces insertions.
Question 2.3 :

1. Quel est le nombre minimum d'insertions d'enregistrements nécessaires à la division de tous les seaux ?

2. Quelle est la valeur de NEXT à l'issue de ces insertions ?
3. Que pouvez-vous dire du nombre de pages composant le quatrième seau après cette série d'insertions ?

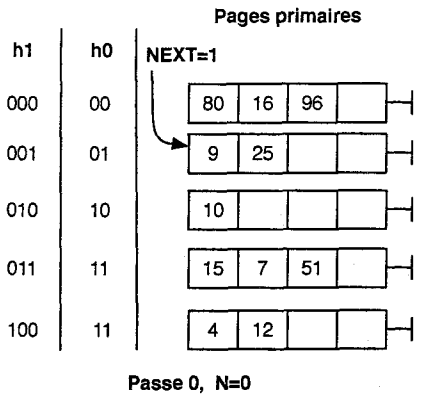

FIG. 2 – Index haché linéaire

Exercice 3 : (Coûts et chemins d'accès) On considère une relation ayant le schéma suivant :

`Employes(eid: integer, ename: string, sal: integer, title: string, age: integer)`

avec `eid` pour clé primaire. Supposons qu'on dispose des index suivants (dont les entrées de données suivent l'alternative 2) : un index haché sur `eid`, un index B+ (non clusterisé) sur `sal`, un index haché sur `age`, et un index B+ clusterisé sur ⟨ `age, sal`⟩. Chaque enregistrement fait 100 octets, et chaque entrée d'index 20 octets. La relation Employés contient 10 000 blocs.
Pour chaque condition qui suit, donner le coût (en nombre d'E/S) du chemin d'accès (i.e. un des index ou un scan) le plus sélectif pour récupérer les enregistrements de Employés qui satisfont la sélection. On supposera que le facteur de réduction de chaque terme est 0.1.

1. `sal>100;`
2. `age=25;`
3. `age>20;`
4. `eid = 1000;`
5. `sal>200 and age>30;`
6. `sal>200 and title='CFO';`
7. `sal>200 and age>30 and title='CFO'.`

Exercice 4 : (Evaluation des requêtes) On considère les relations suivantes :
- `R1(a1: string, b1: string)`
- `R2(a2: string, b2: string)`
- `R3(a3: string, b3: string)`
- `R4(a4: string, b4: string)`

et la requête suivante :

```
SELECT *
FROM R1, R2, R3, R4
WHERE R1.a1=R2.b2 AND R3.a3=R4.b4
```

Chaque table est composée de 320 enregistrements de 128 octets chacun. La taille d'un bloc est de 4096 octets. La sélectivité de la jointure `R1.a1=R2.b2` est 1% (i.e. le ratio de de la taille du résultat de la jointure par la taille du produit cartésien est 1%). La sélectivité de la jointure `R3.a3=R4.b4` est 5% (i.e. le ratio de la taille du résultat de la jointure par la taille du produit cartésien est 5%). Les deux conditions de jointures sont indépendantes. On suppose que la jointure est toujours implantée par boucles imbriquées simple.

Question 4.1 : Traduire la requête précédente dans l'algèbre relationnel.
Question 4.2 : Dessiner le meilleur plan d'évaluation en profondeur gauche pour cette requête.
Question 4.3 : Évaluer le coût (en terme d'E/S) de ce plan.
Question 4.4 : Dessiner le meilleur plan qui ne soit pas nécessairement en profondeur gauche pour la requête.
Question 4.5 : Évaluer le coût de ce plan.

Exercice 5 : (Gestion des transactions) On considère deux séquences d'actions S1 et S2, listées dans l'ordre dans lequel elles sont soumises au SGBD :

S1=`T1:R(A), T2:W(A), T2:W(B), T3:W(B), T1:W(B), T1:commit, T2:commmit, T3:commit`
S2=`T1:R(A), T2:W(B), T2:W(A), T3:W(B), T1:W(B), T1:commit, T2:commmit, T3:commit`

Pour chacune de ces séquences et pour chaque type de contrôle de la concurrence donné ci-dessous, décrivez comment le gestionnaire de transactions traite ces actions. On suppose que les mécanismes d'estampillage attribuent la priorité i à la transaction `T`i. Pour le contrôle par verrouillage, indiquez les requêtes de verrous exclusifs et partagés. Le gestionnaire de transactions traite les actions dans l'ordre donné. Si une transaction est bloquée, on suppose que toutes les actions de cette transaction sont mises dans une file d'attente jusqu'à ce que la transaction soit débloquée ; le gestionnaire de transactions continue alors avec l'action suivante (pour l'ordre de la séquence) d'une transaction non bloquée.

Question 5.1 : 2PL strict.
Question 5.2 : 2PL non-strict.
Question 5.3 : 2PL strict avec prévention des deadlocks par estampillage *Wait-die*.
Question 5.4 : 2PL strict avec prévention des deadlocks par estampillage *Wound-wait*.
Question 5.5 : 2PL strict avec détection des deadlock (donner, le cas échéant le graphe d'attente).