Bases de Données Réparties
Examen du 15 mai 2007

Version CORRIGE

PARTIE 2 avec éléments de correction

Exercice 4 : Optimisation de requêtes réparties 5 pts

Soit **D1**(a, b) sur le site S1, **D2**(a, c) sur S2. La requête R1 = D1 $\bowtie_a$ D2 est posée sur S3. Le schéma du résultat de la requête est (a, b, c). Les attributs sont des entiers positifs. La distribution des valeurs des attributs est uniforme. La taille, en octets, des attributs *a, b, c* est respectivement 100, 400 et 3900 octets.

La sélectivité d'une jointure, notée *s*, est telle que

card(D1 $\bowtie_a$ D2) = s * card(D1) * card(D2)

On note T(D) la taille d'une relation D, exprimée en milliards (10^9) d'octets. On a les valeurs suivantes :

T(D1) = 10 T(D2) = 20 s = 0,001% = 10^{-5}

On considère les plans d'exécution suivants :

P1 : transférer D1 et D2 sur S3 puis traiter la jointure sur S3.
P2 : transférer D1 sur S2, puis traiter la jointure sur S2, puis transférer le résultat sur S3.
P3 : transférer D2 sur S1, puis traiter la jointure sur S1, puis transférer le résultat sur S3.

Question 1

Le coût d'un plan est la somme des transferts de données mesurés en milliards d'octets.

a) Quelle est la cardinalité du résultat de la requête R1 ?

b) Combien vaut T(R1) ?

c) Pour chaque plan P1 à P3, donner son coût.

a) **10^9**

b) **4400 milliards d'octets**

c)

P1 30

P2 4410

P3 4420

d) On suppose que *s* peut varier. Pour quelles valeurs de sélectivité *s*, le plan P1 est-il plus coûteux que P3 ?

***s* est dans l'intervalle [0 ; 2,27 * 10^{-8} [**

Question 2

On considère maintenant que les débits de transfert entre les 3 sites, diffèrent. La fonction *dt(origine,destination)* donne le débit de transfert entre 2 sites, exprimé en milliards d'octets par seconde. On a :

dt(S1, S3) = 0,1 dt(S2, S3) = 1 dt(S1, S2) = dt(S2, S1) = 1

On modifie *s* de telle sorte que la taille du résultat de la requête soit T(R1) = **1** milliard d'octets. On calcule le coût d'un plan comme étant la somme des **temps** de transferts effectués pour traiter la requête.

a) Quelle est la durée du transfert de D1 vers S3, exprimée en secondes ?

100s

b) Quels sont les coûts de P1, P2 et P3 ?

P1 : 120s

P2 : 11s

P3 : 30s

Question 3

On considère maintenant que les débits de transferts peuvent être asymétriques. On a :

dt(S1, S3) = dt(S2, S3) = dt(S1, S2) = 1 dt(S2, S1) = 10

On modifie *s* de telle sorte que la taille du résultat de la requête soit T(R1) = T_0. Pour quelles valeurs de T_0 le plan P3 est-il optimal ?

P1 : 30s P2 : = T_0 + 10 P3 : T_0 + 2

T_0 dans [0,28]

Exercice 5 : JDBC 5 pts

Un moteur M de requêtes réparties est implémenté sous la forme d'un intergiciel (middleware) écrit en java. Il est utilisé pour calculer le résultat d'une élection présidentielle. Chaque département i (i dans [1, 100]) du pays dispose d'une base de données Bi. Toutes les bases sont disjointes.
Le schéma d'une base Bi est le suivant :

InscritDept (n, a)
n est le numéro unique d'électeur
a est l'âge de l'électeur

VoteDept (t, n, c)
t est le tour (1 ou 2),
n le numéro d'électeur
c est formé des initiales des prénom et nom du candidat mentionné sur le bulletin (*c* est une valeur parmi {'NS', 'SR', 'FB', etc.}
Les votes blancs ou nuls ne sont pas répertoriés dans cette relation

M est connecté aux bases Bi à travers l'interface JDBC. M offre à l'application un schéma global représentant les données au niveau national.

InscritNation (n, a, d) // d représente le numéro de département
VoteNation (t, n, c, d)

Question 1

Une application demande à M de traiter la requête R1 permettant de connaître le nombre de votes exprimés au premier tour dans tout le pays, pour chaque candidat, dans l'ordre décroissant du nombre de votes.

Le résultat de R1 est :

c	*nb_votes*
NS	11 448 663
SR	9 500 112
FB	6 820 119
...	
GS	123 540

M traite la requête R1 en minimisant les transferts de données entre les sites. M décompose la requête globale R1 en :

- une requête locale, nommée RL_i, sur chaque base B_i .
- et deux requêtes de recomposition nommées RC1 et RC2.

a) Ecrire en SQL la requête RL_i posée par M sur la base Bi.

RLi : select c, count(*) as nb_votes

From VoteDept

```
Where t=1
Group by c
Order by c
```

b) M recompose les résultats des RLi à l'aide d'un opérateur algébrique *Op* tel que

$RC1 = RL_1$ Op RL_2 Op .... Op RL_{100}.

Quel est l'opérateur Op ?

union

c) Ecrire en SQL la requête RC2 qui permet d'obtenir le résultat de R1 à partir de RC1.

```
RC2 =  select c, sum(nb_votes) as nb_votes
       From RC1
       Group by c
       Order by  nb_votes
```

Question 2

Soit le programme JDBC, nommé P1, affichant le nombre d'inscrits en France en fonction de leur âge.

```
1     String b[] = {"B1", "B2", …, "B100"};
2     Connection c[]  = new Connection[100];
3     Statement s[]  = new Statement[100];
4     ResultSet r[]  = new ResultSet[100];
5     String x = "select a, count(*) from InscritDept group by a order by a";
6     for(int i=0; i<100;i++) {
7        c[i] = DriverManager.getConnection(b[i]);
8        s[i] = c[i].createStatement();
9        r[i] = s[i].executeQuery(x);
10    }
11    while(  Trou 1   ) {
12       int t = r[0].getInt(2);
13       Trou 2;
14       for(int i=1; i<100; i++) {
15          Trou 3;
16          Trou 4 ;
17       }
18      System.out.println( " " + z + "   " + t);
19    }
```

a) Compléter P1. Pour chaque «**Trou**» (marqué en gras), indiquer quelle est l'instruction manquante. Choisir une réponse parmi la liste d'instructions proposées.

Instr 1 : t = r[i].getInt(2)	*Instr 5* : r[0].getInt(2) <= 100
Instr 2 : t += r[i].getInt(2)	*Instr 6* : r[i].next()
Instr 3 : t = t + r[1].getInt(2)	*Instr 7* : r[0].next()
Instr 4 : String z = r[0].getString(1)	*Instr 8* : Autre, préciser

Trou 1 : I7 Trou 2 : I4 Trou 3 : I6 Trou 4 : I2

b) On suppose maintenant que la doyenne de France (Mme Capony agée de 113 ans) s'inscrive sur les listes électorales. Est-ce que P1 donne un résultat correct ? Sinon proposer une modification de P1 pour corriger le problème constaté.

Problème constaté : toutes les requêtes locales RLi n'auront pas la même cardinalité car le domaine de l'attribut age diffère selon les bases. Un seul département possède un age 113.

c) On veut modifier P1 pour que le résultat soit trié par ordre décroissant du nombre d'inscrits. Expliquer brièvement quelles sont les modifications à apporter.

P1 est trié par age croissant

Pour trier par nb d'inscrits il faudrait récupérer toutes les sommes puis les trier en mémoire avant de les afficher

Question 3

On relie M avec une base supplémentaire BS contenant le résultat d'un sondage :

Sondage (d, t, c, p,)
d est le numéro de département dans lequel le sondage a été effectué
t est le tour (1 ou 2)
c est formé des initiales des prénom et nom du candidat
p est la position du candidat dans le sondage (1=1ère position, ...)

On veut que M affiche la liste des numéros de départements pour lesquels le sondage a estimé correctement le candidat arrivé en première position du premier tour dans le département (remarque : le candidat arrivé en tête n'est pas le même dans tous les départements).

a) Combien de requête sont nécessaires pour obtenir le résultat souhaité ?

Réponse : 101 requêtes

b) Si on suppose que le sondage s'avère exact dans 60% des cas. Combien de nuplets doivent être lus par M pour obtenir le résultat souhaité ?

160

c) On considère qu'un nuplet est transféré à chaque appel de la méthode next() retournant la valeur *true* .(i.e., un appel à next() qui retourne *false* ne provoque le transfert d'aucun nuplet). Décrire le traitement de la requête qui minimise le nombre de nuplets transférés.