Nom :	Prénom :	*page 1*

UPMC - UFR 919 - Master d'informatique - M1

UE 4I803 – BDR – 1er examen réparti du 1er mars 2016

Ex1 :	Ex2 :	
Ex3 :	Ex4 :	

Les documents de cours, TD et TME sont autorisés – durée 2h

Tous appareils électroniques éteints et rangés. **Répondre aux questions sur la feuille du sujet** dans les cadres appropriés. Le barème est indicatif. La clarté de la rédaction compte. Ecrire à l'encre bleue ou noire. Ne pas dégrafer le sujet.

Exercice 1 : Index composé et index couvrant une requête — 4 pts

Soit la relation **Joueur** (nujoueur, nom, prénom, âge, ville, sport)

Tous les index sont non-plaçants. Il y a 3 index :

I1 sur Joueur(ville) **I2** sur Joueur**(**âge, sport) **I3** sur Joueur(sport, âge)

Soit les requêtes :

R1 : select sport from Joueur where âge >20

R2 : select max(age) from Joueur where sport = 'vélo'

R3 : select distinct ville from Joueur

R4 : select avg(age) from Joueur where ville='Paris' and sport='vélo'

R5 : select prénom from Joueur where prénom like 'Ted%' and age = 18 and sport = 'judo' order by prénom

a) Est-ce que I2 est utilisable pour R1 ? Entourer `oui` `non.` Justifier.

b) Est-ce que I3 est utilisable pour R1 ? Entourer `oui` `non.` Justifier.

c) Est-ce que I2 est utilisable pour R2 ? Entourer `oui` `non.` Justifier.

d) Est-ce que I3 est utilisable pour R2 ? Entourer `oui` `non.` Justifier.

e) Est-ce que I1 couvre R3 ? Entourer `oui` `non.` Justifier.

f) Peut-on couvrir R4 avec un (ou plusieurs) index parmi ceux existants ? Si oui lesquels ?

g) On sait que R5 peut être évaluée sans accéder à la table Joueur. Expliquer comment évaluer R5 en utilisant I2(âge, sport) et le nouvel index I4(prénom) sans lire aucun nuplet de Joueur.

Exercice 2 : Optimisation de requête et plan d'exécution 8 pts

Soit la base :

Chanteur (<u>nomChanteur</u>, âge, style)
Chanson (<u>titre</u>, nomChanteur, durée, année)
Parole (<u>titre</u>, texte, langue)

On a les index **non** plaçants :

Les clés (soulignées) sont indexées par :	Les autres index sont :
IndNom on Chanteur(nomChanteur), *IndCTitre* on Chanson(titre) *IndPTitre* on Parole (titre).	*IndAge* on Chanteur(age) *IndAnnée* on Chanson(année) *IndLangue* on Parole(langue)

Soit la requête **R1** :

```
Select t.nomChanteur, t.style, p.texte
From Chanteur t, Chanson s, Parole p
Where t.nomChanteur = s.nomChanteur and s.titre = p.titre
and p.langue = 'FR' and t.age = 20 and s.annee = 2016;
```

Soit le plan **P1** exécutant R1 :

```
 ------------------------------------------------------------
| Id  | Operation                          | Name          |
 ------------------------------------------------------------
|   0 | SELECT STATEMENT                   |               |
|   1 |  NESTED LOOPS                      |               |
|   2 |   NESTED LOOPS                     |               |
|   3 |    NESTED LOOPS                    |               |
|*  4 |     TABLE ACCESS FULL              | CHANSON       |
|*  5 |     TABLE ACCESS BY INDEX ROWID    | CHANTEUR      |
|*  6 |      INDEX UNIQUE SCAN             | INDNOM        |
|*  7 |    INDEX UNIQUE SCAN               | INDPTITRE     |
|*  8 |   TABLE ACCESS BY INDEX ROWID      | PAROLE        |
 ------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------
  4 - filter("S"."ANNEE"=2016)
  5 - filter("T"."AGE"=20)
  6 - access("T"."NOMCHANTEUR"="S"."NOMCHANTEUR")
  7 - access("S"."TITRE"="P"."TITRE")
  8 - filter("P"."LANGUE"='FR')
```

Question 1) Compréhension de P1.

a) Dans quel ordre les jointures sont-elles traitées ? Répondre en listant, dans le bon ordre, le nom des relations.

b) La sélection âge=20 est-elle évaluée avant d'évaluer le prédicat de jointure entre Chanteur et Chanson ?

c) P1 utilise-t-il l'index sur Chanson(année) ? Pourquoi ?

d) P1 utilise-t-il l'index sur Parole(langue) ? Pourquoi ?

e) Dessiner l'arbre algébrique linéaire à gauche de R1 tel que les jointures sont dans le **même ordre** que dans P1.Préciser les éventuelles projections qui peuvent être faites sans modifier l'ordre des autres opérations. Inscrire les feuilles de l'arbre sur les traits pointillés en bas du dessin, la racine est en haut.

. . . .

Question 2) Jointure par hachage.

On considère le plan **P2** qui exécute la requête R1 en utilisant seulement des jointures par hachage et autant d'index que possible. Afin de déterminer l'ordre des opérations, on suppose que les paroles françaises ont une taille trop grande pour tenir en mémoire. On suppose que les chanteurs de 20 ans et les chansons de 2016 peuvent tenir en mémoire.

a) Dessiner l'arbre algébrique de la requête telle que les opérations sont dans l'ordre de P2. Ecrire les feuilles de l'arbre sur les traits pointillés, la racine est en haut du dessin.

. . . .

b) Expliquer brièvement les étapes de l'évaluation de P2

c) Quels index peuvent être utilisés pour P2, parmi les index existants ?

Question 3) Coût d'un plan.

Tous les attributs numériques sont entiers. Les hypothèses vues en cours (uniformité, indépendance) sont vérifiées. Le coût est exprimé en nombre de lectures de pages.

Il y a 10 000 chanteurs tenant sur 100 pages. L'âge est dans [11, 60]

Il y a 200 000 chansons tenant sur 1 000 pages. L'année est dans [1917, 2016],

Il y a 200 000 paroles tenant sur 20 000 pages. Il y a 200 langues.

a) Quel est le coût minimal pour la requête **T1** suivante ? select * from Chanteur where âge between 41 and 50 ;

Préciser si l'index (non plaçant) *IndAge* est utilisé ou non.

b) Quel est le coût minimal pour la requête **T2** suivante ? select * from Chanson where année > 2010 ;

Préciser si l'index (non plaçant) *IndAnnée* est utilisé ou non.

c) Quel est le coût minimal pour la requête **T3** suivante ? select * from Parole where langue = 'FR' or langue ='EN' ;

Préciser si l'index (non plaçant) *IndLangue* est utilisé ou non.

d) Quel est le coût de **R2** exécutée en utilisant seulement des jointures par hachage et en supposant qu'on dispose d'un espace mémoire libre de taille infinie ?

R2 :
```
select *
from Chanteur t, Chanson s, Parole p
where t.nomChanteur = s.nomChanteur and s.titre = p.titre
and (p.langue = 'ES' or p.langue = 'RU') and (t.âge between 21 and 30) and s.année > 2000 ;
```

e) Quelle est la taille **minimale** de la mémoire dont on doit disposer, exprimée en nombre de pages, pour exécuter R2 en utilisant seulement des jointures par hachage ?

f) Quel est le coût **minimal** de **R3** en utilisant une jointure par boucles imbriquées avec index ?
R3 : select * from Chanson s, Chanteur t where s.nomChanteur = t.nomChanteur ;

Exercice 3 : Arbres B+ 5 pts

Sauf indication contraire, tous les arbres sont de type arbreB+ d'ordre **1** (i.e., il y a 1 ou 2 valeurs par nœud). On utilise la syntaxe suivante pour représenter un nœud de l'arbre : $N\ (\ v_1, v_2, \ldots)$ où N est le nom du nœud et les v_i sont les valeurs. Quand la feuille F déborde, on garde les **2** plus petites valeurs dans F, la plus grande valeur sera dans la nouvelle feuille. S'il faut choisir une valeur pour un nœud intermédiaire, la choisir, autant que possible, identique à une valeur existant dans une feuille. Toutes les valeurs sont des nombres entiers.

Question 1)

a) Au moment d'insérer une nouvelle valeur, quel est l'inconvénient d'avoir un arbre où tous les nœuds sont déjà remplis?

b) Les trois premières feuilles d'un arbre sont **F1**(9) **F2**(10, 15) **F3**(16). Lorsqu'on insère la nouvelle valeur 13, pourquoi décide-t-on de ne **pas** redistribuer avec F1 ou F3 ?

c) Un arbre a 20 feuilles. Les feuilles et les autres nœuds sont le **plus** remplis possible. Combien de niveaux a l'arbre ? Tenir compte du niveau de la racine et de celui des feuilles. Par exemple, un arbre avec une racine et 3 feuilles a 2 niveaux.

d) Un arbre a 17 feuilles. Les feuilles et les autres nœuds sont le **moins** remplis possible. Combien de niveaux a l'arbre ?

e) Un arbre contient dans ses feuilles les valeurs consécutives {10,11, ... ,26}. Les feuilles et les autres nœuds sont le plus remplis possible. Que contient la racine ?

```
R(...........
```

Question 2) Soit l'arbre A0 composé d'une racine $\mathbf{N_1}(8, 21)$, et des feuilles $\mathbf{F_1}(4)$, $\mathbf{F_2}(10, 13)$ et $\mathbf{F_3}(21)$.

On insère 8. On obtient A1. Dessiner A1.

```
A1 :




                  ....................


Feuilles : F (....................
```

Question 3) Lors d'une suppression, on considère si possible la redistribution à gauche puis à droite, seulement entre des nœuds ayant le même père. L'arbre initial S_0 est composé :

d'une racine R(27)

des valeurs {21, 25, 100} dans les nœuds intermédiaire nommés N_i

et des valeurs {1, 2, 21, 24, 25, 26, 91, 100} dans les feuilles nommées F_j

a) Dessiner S_0.

```
                        R(27)

          ......................................


.............................................................
```

b) On supprime 91 dans S_0. Représenter l'arbre S_1 obtenu (dessiner seulement les nœuds modifiés).

```
S1
```

c) On supprime successivement les valeurs 24 puis 21 dans l'**arbre initial S_0**. Représenter l'arbre S_2 obtenu.

```
S2 (dessiner seulement les nœuds modifiés)

              R(....

       N( ......)  ...

F (........
```

d) On supprime successivement dans l'**arbre initial S_0** les valeurs, dans l'ordre croissant, 1, 2, 21,... jusqu'à ce que l'arbre perdre un niveau. Représenter l'arbre S_3 obtenu.

```
S3
```

Exercice 4 : Table de hachage extensible — 4 pts

Question 1. Dans cette question. Un paquet peut contenir au maximum **2** valeurs (rmq : seulement 2, pas 4).

1) On considère une structure de hachage extensible de profondeur globale PG=3. Le répertoire contient 8 paquets ayant tous une profondeur locale PL=3 : R [P0, P1, P2, P3, P4, P5, P6, P7].

On insère les valeurs 1, 6, 12, 15, 19, 32, 42, 81, 808.

Que contiennent les paquets P0 à P7 ?

```
P0( __  __ ) PL=          P4( __  __ ) PL=
P1( __  __ ) PL=          P5( __  __ ) PL=
P2( __  __ ) PL=          P6( __  __ ) PL=
P3( __  __ ) PL=          P7( __  __ ) PL=
```

2) On considère une table de hachage extensible **T1** de profondeur globale PG=2. Le répertoire contient 4 paquets ayant tous une profondeur locale PL=2 : R [P0, P1, P2, P3].

P0(4, 44) P1(13, 17) P2(6) P3(3)

On supprime 6 de la table **T1** en tentant de fusionner les paquets vides. Quel répertoire obtient-on?

```
R[_ _  _ _  _ _  _ _ ]    PG= _ _
```

Préciser les paquets supprimés et/ou modifiés :

```
Supprimer _ _     et/ou   P_ _ (_ _  _ _)   PL= _ _
```

Question 2. Dans cette question un paquet peut contenir jusqu'à **4 valeurs** au maximum.

On considère la table **T2** avec les paquets A(9) et B(10).

a) Le répertoire est-il R[A, B] ou R[B, A] ? On insère 1, 2, 5, 7, 8, 12 dans T2. Représenter la table **T2'** obtenue.

```
T2' :
Le répertoire est  R[__ , __]   PG=

Paquet  _ _ (_ _  _ _  _ _  _ _)   PL= _ _

Paquet  _ _ (_ _  _ _  _ _  _ _)   PL= _ _
```

b) On insère 13 dans **T2'** obtenue à la question précédente. Représenter la table **T3** obtenue.

```
T3
Le répertoire est R[..., ..., ..............................]

Paquet ...(.............................)  PL=...

Paquet ...( ........................)   PL=...

        ..............................     ......

        ....................................
```

```
...
```

c) On insère 41 dans **T3** obtenue à la question précédente. Représenter la table **T4** obtenue.

```
T4
Le répertoire est R[..., ..., ..............................]

Paquet ...(.............................)  PL=...

Paquet ...( ..........................)   PL=...

         ..............................     ......

         ....................................

...

...
```

Question 3 (bonus).

On a deux tables de hachage H0 et H1 ayant une profondeur PG=1. Chaque table a 2 paquets avec **4 valeurs par paquet**. On veut indexer les nombres pairs dans H0 et impairs dans H1. H0 doit indexer les valeurs {2, 4, 8, 10,12} et H1 les valeurs {1, 5, 9, 11, 19}. Expliquer quelle fonction de hachage utiliser pour qu'on puisse indexer ces valeurs en remplissant les paquets existants, sans créer aucun nouveau paquet.