I54- TD Résolution de requêtes

Partie 1 : bien comprendre

1) Dire qu'une requête est déclarative, c'est dire que (indiquer les phrases correctes) :
   - La requête ne définit pas précisément le résultat.
   - La requête ne dit pas comment calculer le résultat.
   - La requête est indépendante de l'organisation des données.
   - La requête est une expression de besoin en langage naturel.
2) Un plan d'exécution, c'est
   - Un programme choisi parmi un ensemble fixe et pré-défini de programmes proposés par le système.
   - Un programme produit à la volée par le système pour chaque requête.
   - Un arbre d'opérateurs communicants entre eux.
3) L'optimisation de requêtes, c'est
   - Modifier une requête SQL pour qu'elle soit la plus efficace possible.
   - Structurer les données pour qu'elles soient adaptées aux requêtes soumises.
   - Choisir, pour chaque requête, la meilleure manière de l'exécuter.
4) Quelles sont les affirmations vraies parmi les suivantes?
   - Le choix d'un plan d'exécution dépend de la mémoire RAM disponible.
   - Le choix d'un plan d'exécution dépend de la forme de la requête SQL
   - Le choix d'un plan d'exécution dépend de l'existence d'index
   - Le choix d'un plan d'exécution dépend du langage de programmation utilisé.
5) Parmi les requêtes suivantes, quelles sont celles qui nécessitent un opérateur bloquant.
   - `select titre from Film`
   - `select distinct titre from Film`
   - `select count(titre) from Film group by annee`
   - `select titre from Film order by annee`
6) Dans le plan d'exécution de la requête suivante, comment pourrait–on éviter tout parcours séquentiel ?

```
select titre
from   Film f, Rôle r, Artiste a
where  a.nom = 'Stewart' and a.prénom='James'
and    f.id_film = r.id_film
and    r.id_acteur = a.idArtiste
and    f.annee = 1958
```

   - En créant un index sur la clé étrangère *id_film* dans *Rôle*.
   - En créant un index sur le *nom* du rôle dans *Rôle*.
   - En créant un index sur *l'année* de Film.

7) Pourquoi la clé primaire d'une table doit-elle être indexée (plusieurs réponses possibles) :
   - o Parce que la plupart des requêtes SQL portent sur la valeur de la clé primaire.
   - o Pour vérifier rapidement la contrainte d'unicité lors d'une insertion.
   - o Pour vérifier rapidement la contrainte d'intégrité référentielle lors de l'insertion d'une clé étrangère.
   - o Pour vérifier rapidement la contrainte d'intégrité référentielle lors de la destruction d'une clé primaire.

8) Considérons les tables des employés et des départements suivantes :

Enum	*Nom*	*Dnum*
E1	*Benjamin*	*D1*
E2	*Philippe*	*D2*
E3	*Serge*	*D1*

Dnum	*Dnom*
D1	*INRIA*
D2	*CNAM*

   a) Pour une jointure avec index, combien de parcours d'index doit-on effectuer ?
      - o 1
      - o 2
      - o 3

   b) Supposons que l'attribut *Dnum* dans la table Employé soit indexé. Combien de parcours d'index devrait-on effectuer en prenant la table Dept comme table directrice (à gauche).
      - o 1
      - o 2
      - o 3

9) Soit la jointure entre deux tables *T(ABCD)* et *S(MNO)* dans la requête suivante :

```
select * from T, S where T.A=S.M
```

À quels attributs faut-il appliquer la fonction de hachage pour la jointure ?
   - o Aux clés primaires.
   - o Aux attributs A de T et M de S.
   - o À l'attribut A de T, et à la clé primaire de S.
   - o À la clé primaire de T, à l'attribut M de S.

10) Pourquoi 2 nuplets à joindre sont-ils forcément dans des fragments associés ?
   - o Parce que les fragments sont de taille proportionnelle à la table, ce qui garantit l'alignement des nuplets à joindre.
   - o Parce que les valeurs des attributs de jointure étant les mêmes, le résultat de la fonction de hachage est le même.

11) Peut-on appliquer la jointure par hachage à la requête suivante :

```
select * from T, S where T.A <= S.M
```

   - o Oui
   - o Non

Partie 2

On considère la base de données relationnelle suivante :

`employé(noE, nom, prénom, adresse, nodpt)`

`département(no, nom, directeur)`

où `nodpt` est une clé étrangère de la relation `employé` qui réfère la relation `département` et `directeur` (numéro de l'employé directeur du département) est une clé étrangère de la relation `département` qui réfère la relation `employé`.

Les paramètres d'implantation de cette base de données sont:

- card(employé) cardinalité de la relation *employé* 30000
- nb(employé) nombre de blocs occupés par la relation *employé* 3000
- card(département) cardinalité de la relation *département* 100
- nb(département) nombre de blocs occupés par la relation *département* 5
- nv(employé, noE) : nombre de valeurs différentes de l'attribut *noE* dans la relation *employé* 30000
- nv(employé, prénom) nombre de valeurs différentes de l'attribut *prénom* dans la relation *employé* 1000
- nv(employé, adresse) nombre de valeurs différentes de l'attribut *adresse* dans la relation *employé* 30000
- nv(employé, nom) nombre de valeurs différentes de l'attribut *nom* dans la relation *employé* 3000
- nv(département, nom) nombre de valeurs différentes de l'attribut *nom* dans la relation *département* 100
- Index I1 sur la clé primaire de la relation *employé - noE*
- Index I2 sur la clé primaire de la relation *département - no*
- Index groupant I3 sur la clé étrangère *nodpt* de la relation *employé*
- Tampon de 10 cases

Exercice 1

Evaluer le nombre de valeurs différentes des attributs `noE` et `nodpt` de la relation `employé` et des attributs `no` et `directeur` de la relation `département`, en supposant que tout employé travaille dans un et un seul département et ne peut diriger qu'un seul département et qu'un département a un et un seul directeur.

Exercice 2

Evaluer le coût de production en nombre de transferts de pages des opérations suivants :

1. `sel(département, nom = "Informatique")`
2. `sel(employé, nodpt = "125")` dans le cas d'une sélection par boucle et d'une sélection indexée sur l'index I3 ?
3. `join(employé, département, nodpt = no)` dans le cas :
   - d'une jointure par double boucle,
   - d'une jointure par boucle indexée département-index I2,
   - d'une jointure par boucle indexée employé-index I3.

Exercice 3

Pour chacune des deux requêtes suivantes :

(R1)

```
select nom
from employé
where prénom = 'Jean';
```

(R2)

```
select *
from employé,département
where nodpt = no
and directeur = 35;
```

Construire tous les arbres de requêtes équivalents en appliquant les transformations étudiées en cours et en supposant que les opérations algébriques sont la sélection, la projection et la jointure.

Exercice 4

Soit le schéma relationnel suivant :

```
inscription(nom_étudiant, code_ue)
ue(code, sigle_ufr)
responsable(nom_enseignant,code_ue)
ufr(sigle, nom)
```

et la requête suivante :

```
select   nom_étudiant
from     inscription i, ue, responsable r, ufr
where    i.code_ue = ue.code
and       ue.sigle_ufr = ufr.sigle
and       ue.code = r.code_ue
and       r.nom_enseignant = 'Dupont'
and       ufr.nom = 'Lettres‘
```

On suppose que les cardinalités respectives des relations `inscription`, `ue`, `responsable` et `ufr` sont 50000, 300, 450, et 5.

On décide de choisir le « meilleur » arbre de requête en appliquant une heuristique consistant à

1. ne retenir que les arbres de jointure gauche (un arbre dont les opérandes droits des jointures ne contiennent pas de jointure),
2. puis à placer les relations de base par ordre de cardinalité croissante en partant en bas à gauche et en arrivant en haut à droite
3. enfin, en descendant les sélections aussi bas que possible.

Construire cet arbre.

Exercice 5

Soit le schéma relationnel suivant :

```
Journaliste (jid, nom, prénom)
Journal (titre, rédaction, id_rédacteur)
```

La table *Journaliste* stocke les informations (*nom, prénom*) sur les journalistes (*jid* est le numéro d'identification du journaliste). La table *Journal* stocke pour chaque rédaction d'un journal le titre du journal (*titre*), le nom de la rédaction (*rédaction*) et l'id de son rédacteur (*rédacteur_id*).

On a un arbre B+ pour la table Journaliste sur l'attribut *jid*, nommé Idx-Journaliste-jid.

On considère la requête suivante:

```
select nom
from Journal, Journaliste
where jid=id_redacteur
and titre='Le Monde'
and prénom='Jean'
```

Voici deux expressions algébriques:

$\pi_{nom}(\sigma_{titre='LeMonde' \wedge prenom='Jean'}(\text{Journaliste} \bowtie_{jid=redacteur\_id} \text{Journal}))$

et

$\pi_{nom}(\sigma_{prenom='Jean'}(\text{Journaliste}) \bowtie_{jid=redacteur\_id} \sigma_{titre='LeMonde'}(\text{Journal}))$

1) Les deux expressions retournent-elles le même résultat (sont-elles équivalentes) ? Justifiez votre réponse en indiquant les règles de réécriture que l'on peut appliquer.
2) Une expression vous semble-t-elle meilleure que l'autre si on les considère comme des plans d'exécution?
3) Donner le plan d'exécution physique basé sur la jointure par boucles imbriquées indexées, sous forme arborescente.