Durée : 1h00 Le 15/10/2020

Contrôle de rattrapage
Bases de données distribuées

Questions de cours (6 pts)

1. La performance est une condition importante qu'il faut respecter dans la distribution des fragments sur les différents sites de la base distribuée. Qu'est-ce qu'il faut faire pour assurer une bonne performance ?
2. L'application utilisateur ne doit pas être modifiée quand la structure physique des données change **ou** quand la structure logique des données change ?
3. En bases de données distribuées, la réplication (redondance) des données est un inconvénient ou un avantage ?
4. Si on fait une réplication (redondance) de données, est-ce qu'on doit informer l'utilisateur de l'endroit où se trouvent les données répliquées (copiées) ?
5. Un prédicat simple est écrit sous la forme : $p : A_i \ \theta \ val$
   Que représente 'θ' ?
6. Dans le cours, nous avons vu deux manières comment mettre à jour la vue matérialisée ? Citez-les.

Exercice 01 (8 pts)

Un magasin de vente de pièce détachées possède trois points de vente (PV1, PV2, PV3). Le gérant utilise une base de données distribuée pour gérer son stock. La structure de la base est comme suit :

```
PIECE(ref, designation, nbr_en_stock, point_vente)
PRIX(ref, prix_achat)
```

Une pièce possède la même référence dans les trois points de vente.
Une pièce peut être achetée de plusieurs endroits avec des prix différents.
L'attribut 'point_vente' représente l'endroit où est stockée la pièce.

1. Proposez une clé primaire pour chaque relation ?
2. Proposez une décomposition de la base sur les trois sites en utilisant la fragmentation horizontale et/ou verticale ainsi que la réplication des données (précisez le type de la fragmentation utilisée ainsi que la distribution des fragments sur les 3 sites) ?
3. Donnez en SQL la requête répondant à la question : Trouver les références des pièces qui sont mises en vente dans les points de vente 'PV1' et 'PV2'.
4. Le gérant veut limiter l'accès à la base pour un utilisateur de telle sorte à ce qu'il ne peut voir que les références et les désignations des pièces mises en vente dans les points de vente 'PV2' et 'PV3'. Aidez le gérant pour réaliser cette tâche par une requête SQL.

Exercice 02 (6 pts)

Soit la relation Aff (idEmp, idPrj, resp, dur) qui représente l'affectation d'un employé à un projet avec telle responsabilité et telle durée.

Aff

idEmp	idPrj	resp	dur
E1	P1	Manager	12
E2	P1	Analyste	24
E3	P2	Consultant	6
E4	P3	Ingénieur	10
E5	P4	Développeur	48

et soit deux applications qui accèdent à **Aff**. La première application s'exécute sur 5 sites différents {S11, S12, S13, S14, S15} et cherche les durées et les identifiants des employés en utilisant respectivement les prédicats simples suivants :
p11: resp = "Manager" ; p12: resp = "Analyste"; p13: resp = "Consultant" ; p14: resp = "Ingénieur"; p15: resp = "Développeur".

La deuxième application s'exécute sur deux sites différents {S21, S22} et cherche les identifiants des projets et les identifiants des employés en utilisant respectivement les prédicats simples suivants :
P21: dur < 20 ; p22: dur >= 20;

1. Effectuez une fragmentation horizontale de la relation **Aff** en se basant sur les prédicats ci-dessus.
2. Est-ce que cette fragmentation est bonne ? Expliquez pourquoi.

Bonne chance…

NB : Le corrigé type sera disponible sur le site : http://www.larbiguezouli.com

Correction du contrôle de rattrapage
Bases de données distribuées

Questions de cours (6 pts)

1. La performance est une condition importante qu'il faut respecter dans la distribution des fragments sur les différents sites de la base distribuée. Qu'est-ce qu'il faut faire pour assurer une bonne performance ?

Minimiser le temps de réponse ;
Maximiser le débit de chaque site.

2. L'application utilisateur ne doit pas être modifiée quand la structure physique des données change **ou** quand la structure logique des données change ?

L'application utilisateur ne doit pas être modifiée quand la structure **physique** des données change.

3. En bases de données distribuées, la réplication (redondance) des données est un inconvénient ou un avantage ?

En bases de données distribuées, la réplication (redondance) des données est un **avantage**, ça permet de donner une bonne **performance** et de rendre les données **disponibles** en cas de panne.

4. Si on fait une réplication (redondance) de données, est-ce qu'on doit informer l'utilisateur de l'endroit où se trouvent les données répliquées (copiées) ?

L'utilisateur **n'a pas besoin de savoir** si les données sont répliquées ou non. Il envoie ses requêtes comme s'il existe une seule copie des données. (Transparence de réplication).

5. Un prédicat simple est écrit sous la forme : $p : A_i \ \theta \ val$
   Que représente 'θ' ?

$\theta \in \{=, <, >, \neq, \leq, \geq\}$

6. Dans le cours nous avons vu deux manières comment mettre à jour la vue matérialisée ? Citez ces deux manières.

- Le plus simple est de **recalculer** la vue **entière**.
- Une solution **meilleure** est de ne recalculer que la **partie modifiée**.

Exercice 01 (8 pts)

Un magasin de vente de pièce détachées possède trois points de vente (PV1, PV2, PV3). Le gérant utilise une base de données distribuée pour gérer son stock. La structure de la base est comme suit :

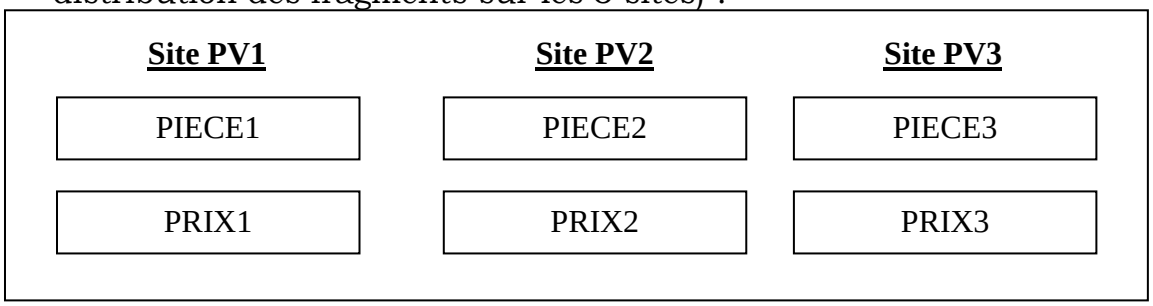
```
PIECE(ref, designation, nbr_en_stock, point_vente)
PRIX(ref, prix_achat)
```

Une pièce possède la même référence dans les trois points de vente.
Une pièce peut être achetée de plusieurs endroits avec des prix différents.
L'attribut 'point_vente' représente l'endroit où est stockée la pièce.

1. Proposez une clé primaire pour chaque relation ?

PIECE : (ref, designation, nbr_en_stock, point_vente) (0.5 pt)
PRIX : (ref, prix_achat) (0.5 pt)

2. Proposez une décomposition de la base sur les trois sites en utilisant la fragmentation horizontale et/ou verticale ainsi que la réplication des données (précisez le type de la fragmentation utilisée ainsi que la distribution des fragments sur les 3 sites) ?



(1 pt)

Fragmentation horizontale de la table PIECE (0.5 pt)

```
CREATE TABLE PIECE1 / PIECE2 / PIECE3 (
ref int NOT NULL,
designation varchar(255) NOT NULL,
nbr_en_stock int,
point_vente varchar(255),
PRIMARY KEY (ref)
) ENGINE=InnoDB

INSERT INTO PIECE1 *
SELECT DISTINCT *
FROM PIECE
WHERE point_vente = 'PV1' (0.5 pt)

INSERT INTO PIECE2 *
SELECT DISTINCT *
FROM PIECE
WHERE point_vente = 'PV2' (0.5 pt)

INSERT INTO PIECE3 *
SELECT DISTINCT *
FROM PIECE
WHERE point_vente = 'PV3' (0.5 pt)
```

Fragmentation dérivée de la table PRIX (0.5 pt)

```
CREATE TABLE PRIX1 / PRIX2 / PRIX3(
ref int NOT NULL,
prix_achat int NOT NULL,
PRIMARY KEY (ref, prix_achat)
) ENGINE=InnoDB

INSERT INTO PRIX1 *
SELECT DISTINCT PRIX.ref, PRIX.prix_achat FROM PIECE1, PRIX
WHERE PIECE1.ref = PRIX.ref (0.5 pt)
```

```
INSERT INTO PRIX2 *
SELECT DISTINCT PRIX.ref, PRIX.prix_achat FROM PIECE2, PRIX
WHERE PIECE2.ref = PRIX.ref (0.5 pt)

INSERT INTO PRIX3 *
SELECT DISTINCT PRIX.ref, PRIX.prix_achat FROM PIECE3, PRIX
WHERE PIECE3.ref = PRIX.ref (0.5 pt)
```

3. Donnez en SQL la requête répondant à la question : Trouver les références des pièces qui sont mises en vente dans les points de vente 'PV1' et 'PV2'.

```
SELECT DISTINCT ref FROM PIECE1
UNION
SELECT DISTINCT ref FROM PIECE2 (1 pt)
```

4. Le gérant veut limiter l'accès à la base pour un utilisateur de telle sorte à ce qu'il ne peut voir que les références et les désignations des pièces mises en vente dans les points de vente 'PV2' et 'PV3'. Aidez le gérant pour réaliser cette tâche par une requête SQL.

```
CREATE VIEW PIECE23(ref, designation)
AS SELECT DISTINCT ref, designation
   FROM PIECE2
   UNION
   SELECT DISTINCT ref, designation
   FROM PIECE3 (1 pt)
```

Exercice 02 (6 pts)

Soit la relation Aff (idEmp, idPrj, resp, dur) qui représente l'affectation d'un employé à un projet avec telle responsabilité et telle durée.

Aff

idEmp	idPrj	resp	dur
E1	P1	Manager	12
E2	P1	Analyste	24
E3	P2	Consultant	6
E4	P3	Ingénieur	10
E5	P4	Développeur	48

et soit deux applications qui accèdent à **Aff**. La première application s'exécute sur 5 sites différents {S11, S12, S13, S14, S15} et cherche les durées et les identifiants des employés en utilisant respectivement les prédicats simples suivants :
p11: resp = "Manager" ; p12: resp = "Analyste"; p13: resp = "Consultant" ; p14: resp = "Ingénieur"; p15: resp = "Développeur".

La deuxième application s'exécute sur deux sites différents {S21, S22} et cherche les identifiants des projets et les identifiants des employés en utilisant respectivement les prédicats simples suivants :
P21: dur < 20 ; p22: dur >= 20;

1. Effectuez une fragmentation horizontale de la relation **Aff** en se basant sur les prédicats ci-dessus.

Aff11 (0.5 pt)

idEmp	dur
E1	12

Aff12 (0.5 pt)

idEmp	dur
E2	24

Aff13 (0.5 pt)

idEmp	dur
E3	6

Aff14 (0.5 pt)

idEmp	dur
E4	10

Aff15 (0.5 pt)

idEmp	dur
E5	48

Aff21 (0.5 pt)

idEmp	idPrj
E1	P1
E3	P2
E4	P3

Aff22 (0.5 pt)

idEmp	idPrj
E2	P1
E5	P4

2. Est-ce que cette fragmentation est bonne ? Expliquez pourquoi.

Cette fragmentation est bonne parce que l'ensemble de prédicats simples P_r={p11, p12, p13, p14, p15, p21, p22} est **complet** et **minimal**. (0.5 pt)
Complet parce que la **probabilité d'accès** par chaque application à n'importe quel tuple des fragments est **égale**. (1 pt)
Minimal parce qu'il y a une **cause** pour effectuer cette fragmentation par les prédicats. Il **existe au-moins** une application qui n'a besoin que d'un seul fragment à un moment donnée. (1 pt)