ESIR1 / BDD mardi 28 février 2012
Contrôle continu 30mn

Répondre de façon concise mais précise aux questions suivantes en utilisant les cadres. Aucun document n'est autorisé, hormis un aide-mémoire de format A4 recto seul constitué par chaque étudiant. Ce sujet comporte 2 pages. Ne pas oublier d'indiquer ses noms et prénoms.

Nom :

Prénom :

Rappeler sans les définir les rôles introduits en base de données.
Administrateur, administrateur d'application, développeur d'application, utilisateur. Des variantes sont admises, mais pas client et serveur, ni persistance et concurrence.

Donner les définitions des termes suivants :

- Equijointure : où le prédicat de jointure est l'égalité.
- Autojointure : où les deux tables sont la même.
- Jointure naturelle : équijointure sur colonnes de même nom avec élimination par projection des colonnes redondantes.

Beaucoup ont recopié sans précaution le transparent qui parle de ça, sans ce rendre compte qu'il s'agissait de notations qui étaient définies dans un autre transparent, du coup réponse sans queue ni tête. Cela confirme que l'aide mémoire ne doit pas être juste une impression au format timbre poste des transparents.

Expliquer les contraintes d'intégrité suivantes :
Là, je voulais les contraintes d'intégrités contenues dans ces termes, pas le sens général de ces termes.

- PRIMARY KEY : unique et not null.
- REFERENCES : existe comme valeur de clé primaire de la table référencées.
- UNIQUE : une seule occurrence dans la colonne, mais plusieurs occurrences possibles dans la table si c'est dans une autre colonne.
- NULL : valeur null, est exclue si NOT NULL
- CHECK prédicat : vérifie que le prédicat est vrai. Pourquoi l'appeler une hypothèse, comme beaucoup l'ont fait ?

Que signifie le D de ACID ? Expliquer.
D pour durabilité. Qui signifie que les effets d'une transaction sont inscrits dans la base de données de façon à résister aux pannes, etc. Cela implique des mécanismes de sauvegarde, journalisation, etc. C'est plus fort que la simple persistance.

Qu'est-ce que la sérialisabilité ?
C'est la propriété d'une exécution concurrente de transactions qui fait le même effet qu'une exécution séquentielle.

Rappeler ce qu'est le protocole à deux phases et son intérêt.
C'est le protocole où dans une transaction, toutes les prises de verrou précèdent toutes les libérations de verrou. Son intérêt est que si toutes les transactions s'y conforment, toute exécution concurrente sera sérialisable. Rien à voir avec interblocage.

Imaginer les verrous qu'il faut poser avant d'exécuter les requêtes SQL suivantes.

SELECT * FROM A, B WHERE A.f = B.g : lockRead de A et B

INSERT INTO A SELECT c, d FROM B : lockRead de B et lockWrite de A

DELETE FROM A WHERE f = 0 : lockWrite de A

Que penser des verrouillages suivants ?

$Lock_S(A)$; a = lire(A) ; a = a+1 ; écrire(A, a) ; Unlock(A) : un verrou shared n'est pas le bon pour protéger une écriture.

$Lock_X(B)$; b = lire(B) ; b = b+1 ; écrire(B, b) ; Unlock(B) ; … ; $Lock_X(B)$;
b = lire(B) ; b = b+1 ; écrire(B, b) ; Unlock(B) : typique d'un verrouillage qui n'isole pas bien un état transitoire de la requête.

$Lock_X(A)$; $Lock_X(B)$; … ; Unlock(A) ; Unlock(B) : un simple verrouillage à deux phases.
Plusieurs remarques bizarres sur le fait que les transactions étant isolées il n'y a pas vraiment besoin de verrouiller, mais on n'en sait rien si elles sont seules !

Quel problème nouveau introduit l'usage des verrous ? Donner une solution simple à exprimer pour prévenir ce problème.
Le nouveau problème est l'interblocage. Une solution simple pour le prévenir est d'acquérir tous les verrous nécessaires en une seule fois, ou que toutes les transactions les acquièrent dans le même ordre. Les méthodes à préemption, Wound-Wait et Wait-Die, ne peuvent pas vraiment passer pour simple !