Aurélien MONNET-PAQUET
Antoine THEBAUD

Compte-rendu TP1 BD

Partie 1

Exercice 1

le rollback a annulé la transaction car celle-ci n'avait pas été commit.

Exercice 2

le rollback n'a pas annulé la transaction car celle-ci avait été commit.

Exercice 4

Le rollback to UnInsert annule seulement le 2e insert ; le rollback "général" annule les 2 transactions.

Partie 2

Exercice 1

1ère variante :
La transaction de décrémation du solde de Claude est refusée. On constate après affichage que le solde de Henri a été mis à jour (250€) mais pas celui de Claude (100€). Le rollback annule la première transaction et remet donc Henri à son ancien solde (200€) .

2ème variante :
La transaction de décrémation du solde de Claude est refusée. On constate après affichage que le solde de Henri a été mis à jour (250€) mais pas celui de Claude (100€). Après validation le solde de Henri a été mis à jour (250€) et Claude a toujours son ancien solde (100€).

3ème variante :
La transaction de décrémation du solde de Claude est acceptée. On constate après affichage que le solde de Henri a été mis à jour (250€) ainsi que celui de Claude (-50€). Le rollback

annule les deux transactions et remet donc Henri et Claude à leurs anciens soldes (respectivement 200€ et 100€) .

4ème variante :
La transaction de décrémation du solde de Claude est acceptée. On constate après affichage que le solde de Henri a été mis à jour (250€) ainsi que celui de Claude (-50€). Lors de la validation la mise à jour du solde de Claude est refusée (violation de contrainte). On constate ensuite avec l'affichage que le 2 transactions ont été annulées, Henri a 200€ et Claude 100€.

Partie 3

Exercice 1

Avant le commit de S1, la modification faite par S1 n'est pas visible par S2. Après le commit de S1, S2 voit les modifications de S1.

Exercice 2

Avant le commit, la modification faite par S1 n'est pas visible par S2. Après commit le de S1, S2 ne voit toujours pas les modifications de S1. Après le commit de S2, S2 voit les modifications de S1.

=> en mode normal, les modifications effectuées dans une première session ne deviennent visibles dans une deuxième session qu'à partir du moment ou la première session a effectué un commit ; en mode sérialisable, les modifications effectuées dans une première session ne deviennent visibles dans une deuxième session qu'à partir du moment ou les sessions ont effectué un commit ;

Exercice 3

3. L'update envoyé par S2 est mise en attente tant que S1 n'a pas commit.
4. Dès que S1 a commit, S2 effectue son update
5. On constate que la table a subit le delete de S1 et ensuite l'update de S2
6. S1 affiche la même chose que S2.

=> On peut voir que le SGBD gère les mises à jour simultanées en mettant en attente les transactions provenant d'une autre session, tant que les mises à jour de la première session n'ont pas été commit.

Exercice 4

=> Dans une situation comme celle-ci la stratégie d'Oracle est de tuer la transaction la plus ancienne