Seyfeddine Jouini

Compte Rendu TP 5

Exercice 1 :

On commence par faire **SET AUTOCOMMIT OFF** pour enlever le commit automatique.

1. On crée une table :

```
CREATE TABLE transaction(
  idTransaction VARCHAR2(44),
  valTransaction NUMBER(10)
);
```

2. On va insérer 2 lignes, faire une modification dans l'une et puis supprimer l'autre.

```
INSERT INTO transaction (idTransaction, valTransaction) VALUES ('TR001', 100);
INSERT INTO transaction (idTransaction, valTransaction) VALUES ('TR002', 200);
UPDATE transaction SET valTransaction = 250 WHERE idTransaction = 'TR001';
DELETE FROM transaction WHERE idTransaction = 'TR002';
SELECT * FROM transaction;
ROLLBACK;
SELECT * FROM transaction;
```

Le premier select :

```
IDTRANSACTION                    VALTRANSACTION
-------------------------------- --------------
TR001                                       250
```

Le deuxième select : Vide .

3. On va faire une insertion de 2 nouvelles lignes et à la fin quite; (RQ: Dans la session S2):

```
INSERT INTO transaction (idTransaction, valTransaction) VALUES ('TR003', 300);
INSERT INTO transaction (idTransaction, valTransaction) VALUES ('TR004', 400);
quit;
```

Le select dans S1 : Donne le vide (ne voie pas les modifications de S2).

4. On va insérer une ligne à partir de S1 :

```
INSERT INTO transaction (idTransaction, valTransaction) VALUES ('TR005', 500);
```

On va faire un shutdown sans commit.

Lors de la reconnexion, en faisant un select on ne voit rien **(autocommit OFF).**

5. On ouvre une 3ème session : S3 :

```
SET AUTOCOMMIT OFF;
INSERT INTO transaction (idTransaction, valTransaction) VALUES ('TR006', 600);
ALTER TABLE transaction ADD (val2transaction NUMBER(10));
ROLLBACK;
SELECT * FROM transaction;
DESCRIBE transaction;
```

On commence par insérer un n-uplet.
On a modifié la structure de la table en ajoutant une colonne val2Trasaction.
Après le rollback :
Le select ne montre rien car toute modification (LMD) est annulée par le rollback (En plus on a paramétré la session par un Set AUTOCOMMIT OFF).
Le describe va montrer la colonne **val2Transaction** car le Rollback annule que les opérations LMD et non pas les DDL (de définition).

6. **Session** : Une session est une connexion spécifique et isolée d'un utilisateur à la base de données, débutant à la connexion et se terminant à la déconnexion.

Transaction : Une transaction est une séquence d'opérations SQL (principalement DML) traitée comme une seule unité logique. Elle garantit l'intégrité des données en assurant soit la réussite complète de toutes les opérations, soit leur annulation totale.

Valider une transaction : Une transaction est validée en utilisant la commande COMMIT. Cela rend les modifications permanentes dans la base de données et visibles pour les autres sessions.

Annuler une transaction : Une transaction est annulée en utilisant la commande ROLLBACK. Cela défait toutes les modifications effectuées depuis le dernier COMMIT (ou le début de la session), restaurant la base de données à son état précédent. Les opérations DDL (comme CREATE TABLE, ALTER TABLE, DROP TABLE) ne sont généralement pas affectées par la commande ROLLBACK.

Exercice 2 :
Dans une première sessions (Session A) : on crée les deux tables ci-dessous :

Seyfeddine Jouini

```sql
CREATE TABLE vol(
  idVol VARCHAR2(44),
  capaciteVol NUMBER(10),
  nbrPlacesReserveesVol NUMBER(10)
);

CREATE TABLE client(
  idClient VARCHAR2(44),
  prenomClient VARCHAR2(11),
  nbrPlacesReserveesCleint NUMBER(10)
);
```

On fait l'insertion de 1 vol et 2 clients :

```sql
INSERT INTO vol (idVol, capaciteVol, nbrPlacesReserveesVol) VALUES ('AF101', 100,
INSERT INTO client (idClient, prenomClient, nbrPlacesReserveesCleint) VALUES ('CL
INSERT INTO client (idClient, prenomClient, nbrPlacesReserveesCleint) VALUES ('CL
```

Isolation des transactions :

A partir de la session A, on va faire une réservation pour le client 1 (CL001).

```sql
UPDATE client
SET nbrPlacesReserveesCleint = nbrPlacesReserveesCleint + 1
WHERE idClient = 'CL001';


UPDATE vol
SET nbrPlacesReserveesVol = nbrPlacesReserveesVol + 1
WHERE idVol = 'AF101';
```

Quand on fait les selects à partir de la session A, on trouve les modifications.

```sql
SELECT * FROM client WHERE idClient = 'CL001';
SELECT * FROM vol WHERE idVol = 'AF101';
```

Session B :

En faisant le select dans un autre session ici B, on constate que les résultats des select dans la session B ne reflètent pas les modifications effectuées dans la Session A (T1). Alice aura toujours le nombre de places réservées initial

COMMIT et ROLLBACK :

Dans la section A, on fait un rollback qui va annuler toutes les modifications :

En faisant un select, on constate qu'on revient à l'état de base comme le montre la figure suivante :
SELECT des clients :

```
IDCLIENT                       PRENOMCLIENT NBRPLACESRESERVEECLEINT
------------------------------ ------------ ----------------------
CL001                          Alice                             0
```

SELECT des vols :

```
IDVOL                          CAPACITEVOL NBRPLACESRESERVEESVOL
------------------------------ ----------- ---------------------
AF101                                  100                     0
```

On va maintenant refaire les modifications et les valider :

```sql
UPDATE client
SET nbrPlacesReserveesCleint = nbrPlacesReserveesCleint + 1
WHERE idClient = 'CL001';
```

```sql
UPDATE vol
SET nbrPlacesReserveesVol = nbrPlacesReserveesVol + 1
WHERE idVol = 'AF101';
```

Et on fait un commit pour écrire les modifications dans les fichiers de la base.
En faisant un rollback, il n y'a rien qui va s'annuler car les modifications sont écrites dans les fichiers de la base (données permanentes).

isolation incomplète = incohérence possible
Réinitialisation de la base :
On supprime le contenu des tables et enfin on réinsère les clients et le vol.

```sql
DELETE FROM client;
DELETE FROM vol;
```

On commence par visualiser le contenu des tables :
Table des vols :

```
CAPACITEVOL NBRPLACESRESERVEESVOL
----------- ---------------------
        100                     0
```

Table des clients : on a séléctionné le nombre de réservations pour chaque client

```
NBRPLACESRESERVEECLEINT
----------------------
                     0
```

Dans transaction T1 : on fait les mises à jours suivants :

```
UPDATE client
SET nbrPlacesReserveesCleint = nbrPlacesReserveesCleint + 3
WHERE idClient = 'CL002';
```

```
UPDATE vol
SET nbrPlacesReserveesVol = nbrPlacesReserveesVol + 3
WHERE idVol = 'AF101';
```

Et dans la T2 :

```
UPDATE client
SET nbrPlacesReserveesCleint = nbrPlacesReserveesCleint + 2
WHERE idClient = 'CL001';
```

```
UPDATE vol
SET nbrPlacesReserveesVol = nbrPlacesReserveesVol + 2
WHERE idVol = 'AF101';
```

Selection des clients :

```
IDCLIENT                       PRENOMCLIENT NBRPLACESRESERVEECLEINT
------------------------------ ------------ ----------------------
CL001                          Alice                             2
CL002                          Bob                               3
```

On remarque que tout est bon.
Cependant, quand on sélectionne les vols, on constate une mise à jour perdue:

```
IDVOL                           CAPACITEVOL NBRPLACESRESERVEESVOL
------------------------------- ----------- ---------------------
AF101                                   100                     3
```