

Bases de Données Avancées

septembre 2010

TP – transactions

Vous allez travailler par deux connectés simultanément sur deux machines voisines¹. Les manipulations qui suivent utiliseront une table T ayant 2 colonnes numériques A et B.

Par la suite, j'utiliserai les abbréviations :

- `select T` pour `select * from T` ;
- `select T(A = 1)` pour `select * from T where A=1` ;
- `insert (3,6)` pour `insert into T values (3,6)` ;
- `update(A <- A+1)` pour `update T set A = A+1` ;
- `update(A <- 3)|(A = 4)` pour `update T set A=3 where A=4` ;
- `delete(A = 4)` pour `delete from T where A=4` ;

Pour commencer, créez la table T (une par binôme) et rendez-la accessible aux deux expérimentateurs (instruction GRANT) :

session 1
BDAFCX

session 2
BDAFCY

create table T(
 a number,
 b number);

grant select, insert,
 delete, update
 on T
 to BDAFCY ;

create synonym T
 for BDAFCX.T

Insérez les lignes (0,0), (2,3) et (0,1), et faites un `commit` juste après l'insertion :

session 1

insert (0,0)
insert (2,3)
insert (0,1)
commit

Transactions standards

Question 1 :

session 1

insert (4,4)
insert ('a','b')
select T

¹vous pouvez aussi réaliser ce TP seul, en ouvrant deux connexions à SQLPlus

Quelles instructions sont annulées ?

```
      session 1
-----
insert (5,5)
rollback
select T
```

Et maintenant ?

Question 2 :

session 1	session 2

select T	
	update(A <- A+1)
select T	
	commit
select T	
commit	

Quel problème est mis en évidence ?

Question 3 :

session 1	session 2

update(A <- A+1)	
select T	
	insert(3,7)
	select T
	delete(A=2)
	delete(A=1)
commit	
	commit

L'ordonnancement de ces deux transactions est-il sérialisable, c'est à dire équivalent à une exécution en série ?

Question 4 :

session 1	session 2
update(A<-3) (A=2)	
	update(B<-2) (B=3)
select T	select T
	update(A<-3) (A=2)
update(B<-2) (B=3)	
commit	commit

Quels sont les verrous posés par **session 1** et **session 2** avant le deuxième **update** de **session 2** ?
Comment se terminent ces transactions ?

Question 5 : Avec Oracle, il existe une clause **for update** à l'instruction **select**. L'expérience suivante va vous permettre de comprendre son utilité (on reviendra sur cette clause quand on abordera de nouvelles spécificités de PL/SQL).

session 1	session 2
select T(B=7) for update	
	update(B<-10) (B=0)
	select T ;
	update(A<-5) (A=3)
update(B<-6) (B=7) ;	
commit	
	commit
	select T

Quel type de verrouillage est effectué ?

Le mode Read-Only

Question 6 :

session 1	session 2
set transaction read only;	
select T	
	select T
	update(A<-A+1)
	commit
select T	
	select T
	update(A<-A+1)
	commit
select T	
commit	
select T	

Quel problème est résolu ? Comment est-il résolu ?

Verrouillages explicites

Question 7 :

session 1	session 2
lock table T in share mode ;	
commit	select T for update
	commit
lock table T in exclusive mode ;	select T for update
commit	commit

Comment expliquez-vous ce résultat ?

Les instructions du DDL

Question 8 :

session 1 BDAFCX	session 2 BDAFCY
create synonym S for T	
grant select on S to BDAFCY	
	select BDAFCX.S
drop synonym S	
	select BDAFCX.S
rollback	
select S	

Comment se comportent les instructions du DDL ?

Transactions en PL/SQL

Activez l'affichage dans SQLPlus grâce à la commande `SET SERVEROUTPUT ON`.

Supprimez toutes les données de T et ajoutez une contrainte de clé primaire :

```
alter table T add constraint T_Pkey primary key(a);
```

Pour toutes les questions suivantes, vous regarderez le contenu de la table T après exécution du code PL/SQL et vous en déduirez ce qui s'est produit.

Question 9 :

Exécutez le bloc anonyme suivant :

```
begin
  insert into T values (-1,-1) ;
  savepoint p1 ;
```

```
insert into T values (-2,-2) ;
rollback to p1 ;
commit ;
end ;
```

Question 10 :

Créez la procédure p suivante :

```
create or replace procedure p(i1 NUMBER, i2 NUMBER) is
begin
insert into T values (i1,i2);
insert into T values (i1,i2+1);
insert into T values (i1+1,i2+1);
commit ;
end ;
```

Exécutez cette procédure par la commande `execute p(50,50)`.

Question 11 :

Modifiez la procédure p en ajoutant un traitement d'exception :

```
create or replace procedure p(i1 NUMBER, i2 NUMBER) is
begin
insert into T values (i1,i2);
insert into T values (i1,i2+1);
insert into T values (i1+1,i2+1);
commit ;
exception
when others then dbms_output.put_line('pb insertion'); commit;
end ;
```

Exécutez à nouveau cette procédure par la commande `execute p(50,50)`.

Question 12 : Définissez les procédures p et pp suivantes :

```
create or replace procedure pp(i1 NUMBER, i2 NUMBER) is
begin
insert into T values (i1,i2);
exception
when others then dbms_output.put_line('pb insertion'); commit;
end ;
```

```
create or replace procedure p(i1 NUMBER, i2 NUMBER) is
begin
insert into T values (i1,i2);
pp(i1,i2);
insert into T values (i1+1,i2+2);
commit ;
exception
when others then dbms_output.put_line('pb insertion'); commit;
end ;
```

Exécutez `p(60,60)`

Question 13 :

```
create or replace procedure pp(i1 NUMBER, i2 NUMBER) is
begin
    insert into T values (i1,i2);
    insert into T values (i1+1,i2+1);
exception
    when others then dbms_output.put_line('pb insertion pp');
end ;
```

```
create or replace procedure p(i1 NUMBER, i2 NUMBER) is
begin
    insert into T values (i1,i2);
    pp(i1+1,i2);
    insert into T values (i1+1,i2+2);
    commit ;
exception
    when others then dbms_output.put_line('pb insertion p');
                    rollback;
end ;
```

Exécutez p(70,70).

Dans les questions suivantes, p reste inchangée.

Question 14 : Remplacez pp par cette procédure (on ajoute un commit) :

```
create or replace procedure pp(i1 NUMBER, i2 NUMBER) is
begin
    insert into T values (i1,i2);
    insert into T values (i1+1,i2+1);
    commit ;
exception
    when others then dbms_output.put_line('pb insertion pp');
end ;
```

Exécutez p(80,80).

Question 15 : Remplacez pp par cette procédure qui utilise une transaction autonome :

```
create or replace procedure pp(i1 NUMBER, i2 NUMBER) is
    pragma autonomous_transaction ;
begin
    insert into T values (i1,i2);
    insert into T values (i1+1,i2+1);
    commit ;
exception
    when others then dbms_output.put_line('pb insertion pp');
end ;
```

Exécutez p(90,90);