

T.D 2.

Gestion de la concurrence d'accès

Dans les questions qui suivent les deux opérations find O et upd O, désignent respectivement une opération de lecture et de mise à jour du granule O.

Exercice 1 Indiquer pour chacune de ces questions le résultat de l'exécution des transactions T1 et T2 (sérialisable, perte de mise à jour, etc.) .

a) T1:find S;T1:find C1;T1:upd S;T1:upd C1;T2:find S;T2:find C2;T2:upd S;T2:upd C2

b) T1:find S;T1:find C1;T2:find S;T2:find C2;T2:upd S;T2:upd C2;T1:upd S;T1:upd C1

c) T1:find S;T1:find C1;T1:upd S;T2:find S;T2:find C2;T2:upd S;T1:upd C1;T2:upd C2

d) T1:find S;T1:find C1;T1:upd S;T2:find S; T2:find C2;T2:upd S;T2:upd C2;T2:commit;T1:upd C1;T1:Rollback

e) T1:find S;T2:find S;T1:upd S ;T2:find C2;T2:find S; T1:Commit;T2:upd C2;T2:Commit

Exercice 2 Nous considérons une séquence d'opérations impliquant des transactions T1,T2,T3,T4 où A et B sont deux granules de la base de données. On suppose que T1,T2,T3 et T4 suivent respectivement les protocoles PSE, PS, PX et PUE (vus en cours). Compléter le tableau suivant en indiquant à chaque instant t l'état de la pile et le graphe d'attente entre transactions.

T1 (PSE)	T2 (PS)	T3 (PX)	T4 (PUE)	Temps	Pile verrous	Graphe d'attente
D (Début)				t1		
	D			t2		
			D	t3		
Find (A)				t4		
			Find (B)	t5		
	Find (B)			t6		
		D		t7		
	Find(A)			t8		

			Upd(B)	t9		
		Find (B)		t10		
Find (B)				t11		
	Upd(B)			t12		
			Commit	t13		
	commit			t14		
		Find(A)		t15		
Commit				t16		
		Commit		t17		

Exercice 3 On considère le tableau de l'exercice précédent. Compléter ce tableau en adoptant une approche préventive (Wait-Die / Wound-Wait). Comparer le résultat de ces deux approches.

Thème : Mise en oeuvre sous Oracle

Exercice 4 Vous ouvrez deux fenêtres SQLDeveloper pour avoir deux transactions concurrentes. Vous appelez vos connexions *fenetre1* et *fenetre2*. Vous faites ce qui est indiqué ci-dessous. Vous expliquez ce qui se passe et vous justifiez vos réponses. **Les questions sont indépendantes : les modifications apportées à la base dans une question ne sont pas prises en compte dans les autres questions (sauf indication).**

1)

fenetre1

select * from employe;

select * from employe;

select * from employe;

fenetre2

insert into employe values (99,...);

select * from employe;

commit;

2)

fenetre1

select * from employe;

select * from employe;

select * from employe;

fenetre2

delete from employe where nuempl=99

select * from employe;

commit;

3)

fenetre1

set transaction read only

select * from employe;

select * from employe;

commit;

fenetre2

insert into employe values (99,...);

select * from employe;

commit ;

4)

fenetre1

insert into employe values(99,...)

commit;

select * from employe;

fenetre2

update employe set ... where nuempl=99;

commit;

select * from employe;

5) On suppose que l'employé 99 est dans la table employé

fenetre1

delete from employe where nuempl=99;

rollback;

select * from employe;

fenetre2

update employe set ... where nuempl=99;

select * from employe;

6) On ajoute une contrainte d'intégrité référentielle (FKEmploye) entre nuempl dans la table travail et nuempl dans la table employe. On suppose que l'employé 99 est dans la table employé

Fenetre1

delete from employe where nuempl=99;

select * from travail where nuempl=99;

commit;

fenetre2

insert into travail values (99,135,5);

7)

fenetre1

insert into employe values(100,'Martin',22,3);

select * from employe;

savepoint p1;

insert into employe values(101,'Dupond',25,3);

select * from employe;

rollback to p1;

select * from employe;

commit;

fenetre2

update employe set nomempl='Dupond' where nuempl=101;

8) les employés numéros 100 et 101 existent dans la table employé.

fenetre1

select * from employe where nuempl=100 for update

update employe set nomempl='Marcel' where nuempl=100;

commit;

fenetre2

select * from employe where nuempl=100 for update

update employe set nomempl='Martin' where nuempl=100;

commit;

9) les employés numéros 100 et 101 existent dans la table employé.

fenetre1

select * from employe where nuempl=100 for update

update employe set nomempl='Martin' where nuempl=100;

commit ;

fenetre2

select * from employe where nuempl=101 for update

update employe set nomempl='Marcel' where nuempl=101;

commit;

10)

fenetre1

lock table employe in share mode;

insert into employe values(101,'Dupond',2,5);

commit ;

fenetre2

insert into employe values(100,'Martin',22,3);

commit

11)

fenetre1

lock table employe in share mode;

commit ;

fenetre2

insert into travail values(100,2,23);

12)

fenetre1

select * from employe where nuempl=100;

insert into travail values(100,2,23);

fenetre2

lock table employe in share mode;

lock table travail in share mode

insert into travail values(101,2,23);

commit ;

13)

fenetre1

lock table employe in share mode;

update employe set nomempl='Martin'
where nuempl=101;

commit;

fenetre2

lock table employe in row share mode;

update employe set nomempl='Dupont'
where nuempl=99;

14)

fenetre1

lock table employe in row share mode;

update employe set nomempl='Martin'
where nuempl=101;

commit;

fenetre2

lock table employe in row share mode;

update employe set nomempl='Dupont'
where nuempl=100;

commit ;

15)

fenetre1

lock table employe in exclusive mode;

update employe set nomempl='Martin'
where nuempl=101;
commit;

fenetre2

lock table employe in row share mode;

16)

fenetre1

select * from employe where nuempl=101 for update;

update employe set nomempl='Martin' where nuempl=101;
commit;

fenetre2

select * from employe where nuempl=101 for update;

update employe set nomempl='Dupont' where nuempl=101;
commit;

17) les employés 100 et 101 existent dans la base de données.

fenetre1

lock table employe in row exclusive mode;

update employe set nomempl='test' where nuempl=101;

update employe set nomempl='test2' where nuempl=100;

commit ;

fenetre2

lock table employe in row share mode;

update employe set nomempl='test' where nuempl=100;

commit;

18)

Ajouter dans la table service un attribut *nbreempl* de type *number*. Pour chaque service cet attribut prend comme valeur le nombre d' employés affectés au service.

Ecrire un trigger *trig-nbreempl* qui incrémente l'attribut ***nbreempl*** après l'ajout d'un employé.

19)

fenetre1

insert into employe values(99,...,2)
commit

20)

fenetre1

select nbreempl from service where nuserv=2

insert into employe values(99,...,2)

select nbreempl from service where nuserv=2
commit

fenetre2

set transaction read only
select nbreempl from service where nuserv=2

select nbreempl from service where nuserv=2

fenetre2

select nbreempl from service where nuserv=2

insert into employe values(100,...,2)

select nbreempl from service where nuserv=2
commit ;