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). Complétez 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**		
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		

Exercice 3 On considère deux transactions T2 et T1 avec ti et tj (i < j) représentant respectivement le temps de démarrage de T1 et T2. Supposons que T1 souhaite accéder à la table R qui est déjà verrouillée exclusivement par T2.

Expliquez le comportement de T1 et T2 si on considère le protocole Wait-Die ?

Thème : *Mise en oeuvre sous Oracle*

Exercice 4 *Vous ouvrez deux fenêtres SQLDeveloppper pour avoir deux transactions concurrentes. Vous appelez vos connexions fenetre1et 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	***fenetre2***
select * from employe;	
	insert into employe values (99,...);
select * from employe;	
	select * from employe;
	commit;
select * from employe;	

2)

fenetre1	***fenetre2***
select * from employe;	
	delete from employe where nuempl=99
	select * from employe;
select * from employe;	
	commit;
select * from employe;	

3)

fenetre1	***fenetre2***
set transaction read only	
select * from employe;	
	insert into employe values (99,...);
select * from employe;	
commit;	
	select * from employe;

4)

fenetre1	***fenetre2***
insert into employe values(99,...)	
	update employe set ... where nuempl=99;
commit;	

select * from employe;

commit;

select * from employe;

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

fenetre1 ***fenetre2***

delete from employe where nuempl=99;

update employe set ... where nuempl=99;

rollback;
select * from employe;

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 ***fenetre2***

delete from employe where nuempl=99;

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

select * from travail where nuempl=99;

commit;

7)

fenetre1 ***fenetre2***

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;

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

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

fenetre1 ***fenetre2***

select * from employe where nuempl=100 for update

select * from employe where nuempl=100 for update

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

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

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

fenetre1 ***fenetre2***

select * from employe where nuempl=100 for update

select * from employe where nuempl=101 for update

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

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

commit ;

commit;

10)

fenetre1 ***fenetre2***

lock table employe in share mode;

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

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

commit ;

commit

11)

fenetre1 ***fenetre2***

lock table employe in share mode;

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

commit ;

12)

fenetre1 ***fenetre2***

select * from employe where nuempl=100;

lock table employe in share mode;

lock table travail in share mode

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

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

commit ;

13)

fenetre1 **fenetre2**

lock table employe in share mode;

lock table employe in row share mode;

update employe set nomempl='Martin'

where nuempl=101;

update employe set nomempl='Dupont'

where nuempl=99;

commit;

14)

fenetre1 ***fenetre2***

lock table emplye in row share mode;

lock table employe in row share mode;

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

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

commit;

commit ;

15)

fenetre1 ***fenetre2***

lock table employe in exclusive mode;

lock table employe in row share mode;

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

16)

fenetre1 ***fenetre2***

select * from employe where nuempl=101 for update;

select * from employe where nuempl=101 for update;

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

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

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

fenetre1 **fenetre2**

lock table employe in row exclusive mode;

lock table employe in row share mode;

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

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

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

commit;

commit ;

18)

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

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

19)

fenetre1	***fenetre2***
	set transaction read only *select nbrempl from service where nuserv=2*
insert into employe values(99,…,2) *commit*	
	select nbrempl from service where nuserv=2

20)

fenetre1	***fenetre2***
select nbrempl from service where nuserv=2	
	select nbrempl from service where nuserv=2
insert into employe values(99,…,2)	
	insert into employe values(100,…,2)
select nbrempl from service where nuserv=2 *commit*	
	select nbrempl from service where nuserv=2 commit ;