HOMEWORK-9

NIHAR MUNIRAJU.

Q1. Transactions- Concurrency

S: UPDATE TEST SET X:= X +10 WHERE AID=1; UPDATE TEST SET Y: Y-10 WHERE AID=1;

T: UPDATE TEST SET X:=X*2 WHERE AID=1; UPDATE TEST SET Y:=Y*2 WHERE AID=1;

U: UPDATE TEST SET Y: = Y+10 WHERE AID =1; UPDATE TEST SET X :=X-10 WHERE AID=1;

Assuming initial values of X=15 and Y=25, concurrent execution of these three transcations can leave the database in various states. Determine the state of the database (values of X & Y) assuming isolaton level serializable for each S, T, U.

Soln

⇒ First let's run S having X=15 & Y= 25, AID=1.

Since X: X+10.
X: 15+10
X = 25

Y = 25.
Y = Y-10
Y = 25-10
Y = 15

$\boxed{\text{S will have } X=25, Y=15}$.

⇒ Lets take T where the case is X=X*2, Y=Y*2, AID=1.

Since X=X*2
From S[X=25, Y=15].
X= 25*2
X = 50.

Y= Y*2.
Y=15*2
Y = 30.

T will have $\boxed{X=50, Y=30}$.

⇒ Lets take U where the case is. X=X-10, Y=Y+10 & AID=1.

Since X=50 & Y=30 from T.

X = X-10
X = 50-10
X=40.

Y=Y+10
Y= 30+10
Y=40.

U will have $\boxed{X=40, Y=40.}$

2). Transactions - Representation.

Consider table Item(name, price) where name is a key, and the following two concurrent transcitions.

T1:

Begin Transcation;

S1: Insert into Values ('FCDB', 40);

S2: Update Item Set price = price +30 Where name = 'EN';

Commit;

T2:

Begin Transcation;

S3: Select Avg(price) As a1 From Item;

S4: Select Max(price) As a2 From Item;

Commit;

Soln

T1:

Begin

Read item

Write item ('FCDB', 40)

Read Price

A.Price = A.Price +30.

Write A.Price.

Commit.

T2:

Begin

Read item (price).

A. Price = A.Price1+ A.Price2+...+A.Pricen

A1 = A.Price/n

Read A1.

Read item (price)

A2 = max(price)

Read A2

Commit.

Q3 - Transactions - Weaker Isolation levels.

Soln

a) Yes, Nonserializable behaviour is possible for this as first the Statements from the transaction 1 is executed and then the statements in trahscation 2 will be executed.

→ The first transcation will report values R:0 & S=0 .

The second transcation will report values R:1 & S=1.

→ The first transcation, the read uncommitted isolation levels allows the first SELECT statement to read any data that has been Commited by other transcations, even if that data is part of an uncommitted transcaitions. This means that the first transcition will see the data from the second transcition which has not yet been Committed. The Second transition will see the data from the first transcition; which has already been committed.

b) No, nonserializable behaviour is not possible because here in transaction 2 only R values in being updated but the result of second statement in transition 1 which belongsto relation S is same before and after the executation of transcation 2.

→ The first transcation will report values R:2 & S= 1 .

→ The Second transcation will report values R:2 & S=2 .

c) Yes, nonserializable behaviour is possible in this first and third third statements of the transcation 1 is executed before the transaction 2 and after the transcation 2 .

→ The first transcition will report values R:0 & S:0 & R:0

→ The second transocition will report values R:1 & S:0 .

→ The first transacition will not see the data from the second transcations, which has not yet been committed. The second transcations will see the data from the first transcitions, which may already been committed.