Converson Deadlock:

Test case 1:

Mô tả tình huống tranh chấp:

Giả sử khi đơn hàng có mã *X* xuất một kho (Ví dụ tên kho là: Mê linh, Hà Nội), lúc này đối tác tìm đơn hàng có mã đơn hàng là *X*  đó để kiểm tra đơn hàng có trong kho không, sau đó đối tác sẽ cập nhật tình trạng đơn hàng *X*  thành “Đơn hàng đã xuất kho Mê Linh, Hà Nội”. Tuy nhiên cùng lúc đó nhận hàng, thì tài xế sẽ đọc lên thông tin giao hàng để lên kế hoạch đi giao đồng thời cập nhật tình trạng đơn hàng thành “Tài xế nhận hàng”.

|  |  |  |
| --- | --- | --- |
| T1 \_ Đối tác | T2 \_ Tài xế | Ghi Chú |
| SET TRAN ISOLATION LEVEL SERIALIZABLE | | Gán mức cô lập Serializable cho cả hai giao tác. |
| Shared\_Lock(DON\_HANG) |  | Đối tác thành công xin khóa đọc trên bảng đơn hàng vì bảng đó chưa có khóa nào chiếm. |
| IF EXISTS (Select \* From DON\_HANG Where MaDT = @madt AND MaDH = @madh); |  | Đối tác đọc trên bảng đơn hàng thành công vì đã có đã chiếm được shared lock trước đó. |
|  | Shared\_lock(DON\_HANG) | Tài xế thành công xin khóa đọc trên bảng đơn hàng. Vì bảng đơn hàng đã bị chiếm với khóa shared lock nên có thể chiếm giữ khóa S cùng lúc được. |
|  | Select TenDuong From DON\_HANG Where MaDH = @madh | Tài xế đọc đơn hàng thành công vì đã chiếm khóa đọc trước đó. |
|  | Exclusive\_Lock(DON\_HANG)  UPDATE DON\_HANG (TinhTrangDH)  SET TinhTrangDH = “Tài xế đã nhận đơn hàng”  WHERE MaDH = @madh | Tài xế không thể xin được khóa ghi vì bảng đơn hàng đã bị chiếm khóa đọc trước bởi T1 và cả T2 trước đó. Mà do 2 transaction đều đang ở mức cô lập Serializable nên hai khóa đều được giữ đến cuối giao tác. Lúc này giao tác 2 chưa hoàn thành nên chưa trả khóa đọc. |
| Exclusive\_Lock(DON\_HANG)  UPDATE DON\_HANG (TinhTrangDH)  SET TinhTrangDH = “Đã xuất kho Mê Linh Hà Nội”  WHERE MaDH = @madh |  | Đối tác cũng không thể cập nhật được tình trạng đơn hàng vì không xin được khóa ghi do bảng đơn hàng đã bị chiếm bởi khóa đọc. |

Cycle Deadlock:

TEST CASE 2:

Mô tả tình huống tranh chấp: