Câu 1

a)

CREATE PROCEDURE GetPurchaseOrderTotal

@SOPHIEUDH INT

AS

BEGIN

SELECT SUM(DONGGIA \* SLDAT) AS TONGTIEN

FROM CTDATHANG

WHERE SOPHIEUDH = @SOPHIEUDH

END

b)

CREATE TRIGGER CheckSLGIAO

ON CTGIAOHANG

AFTER INSERT, UPDATE

AS

BEGIN

IF EXISTS (

SELECT 1

FROM inserted i

INNER JOIN CTDATHANG c ON i.MAHH = c.MAHH AND i.SOPHIEUDH = c.SOPHIEUDH

WHERE i.SLGIAO > c.SLDAT

)

BEGIN

RAISERROR(Số lượng hàng giao không được lớn hơn số lượng hàng đặt ', 16, 1)

ROLLBACK TRANSACTION

RETURN

END

END

c)

SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;

GO

CREATE PROCEDURE [dbo].[usp\_InsertOrUpdate\_HANGHOA]

@MAHH NVARCHAR(10),

@TENHH NVARCHAR(50),

@DVT NVARCHAR(10),

@SLTON INT

AS

BEGIN

BEGIN TRANSACTION;

WAITFOR DELAY '00:00:10'; -- Chèn lệnh delay để mô phỏng thời gian thực thi của giao dịch

IF EXISTS (SELECT \* FROM HANGHOA WHERE MAHH = @MAHH)

BEGIN

UPDATE HANGHOA SET TENHH = @TENHH, DVT = @DVT, SLTON = @SLTON WHERE MAHH = @MAHH;

END

ELSE

BEGIN

INSERT INTO HANGHOA (MAHH, TENHH, DVT, SLTON) VALUES (@MAHH, @TENHH, @DVT, @SLTON);

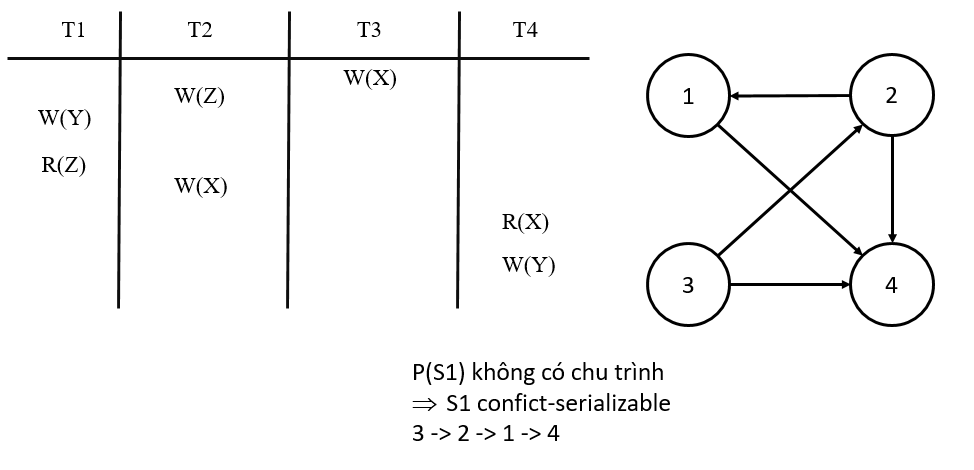
END

COMMIT TRANSACTION;

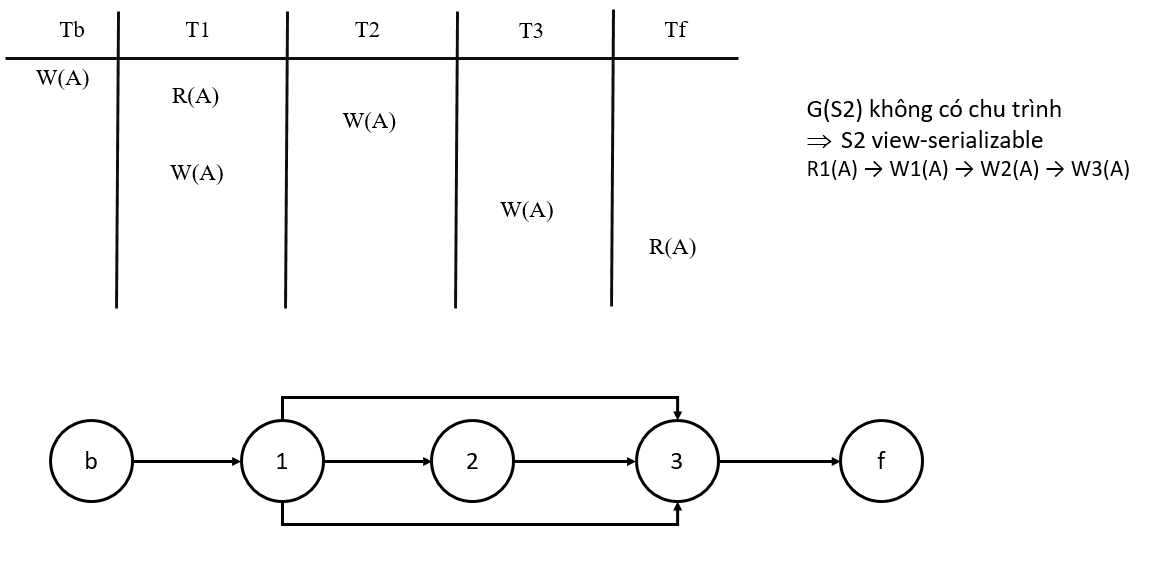
END

Câu 2 :

a)



b)



Câu 3 :

A: được sử dụng bởi giao dịch 1 và 3.

B: được sử dụng bởi giao dịch 2 và 4.

C: được sử dụng bởi giao dịch 3 và 2.

Các giao dịch sử dụng tài nguyên theo thứ tự như sau:

* Giao dịch 2 yêu cầu khóa tài nguyên B.
* Giao dịch 1 yêu cầu khóa tài nguyên A.
* Giao dịch 4 yêu cầu khóa tài nguyên B.
* Giao dịch 3 yêu cầu khóa tài nguyên C.

Tuy nhiên, các giao dịch cũng yêu cầu khóa tài nguyên khác trước khi giải phóng tài nguyên đã khóa. Điều này dẫn đến tình huống khóa chết (deadlock) nếu các tài nguyên không được giải phóng đúng thứ tự.

Giả sử rằng giao dịch 2 đã khóa tài nguyên B trước khi giao dịch 1 khóa tài nguyên A. Tiếp theo, giao dịch 1 sẽ yêu cầu khóa tài nguyên B, nhưng tài nguyên B đã được khóa bởi giao dịch 2, dẫn đến deadlock. Tương tự, giao dịch 4 cũng yêu cầu khóa tài nguyên B, nhưng tài nguyên này đã được khóa bởi giao dịch 2.

🡪 Vì vậy, lịch trình trên sẽ dẫn đến deadlock nếu các tài nguyên không được giải phóng đúng thứ tự.