**d**

**BÁO CÁO ĐỒ ÁN THỰC HÀNH ĐỢT 3**

**MÔN HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU**

GVHD:

GV. Hồ Thị Hoàng Vy

GV. Tiết Gia Hồng

# **THÔNG TIN NHÓM**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **MSSV** | **Họ tên** | **Công việc** | **% Hoàn thành** |
| **1** | 18127022 | Phạm Ngọc Thùy Trang | * Viết sườn báo cáo cho các thành viên và chỉnh sửa đợt 1 * Viết và code mô phỏng 4 tình huống tranh chấp lỗi đồng thời * Chỉnh sửa báo cáo đợt cuối | **100%** |
| **2** | 18127053 | Ngô Thị Thanh Thảo | * Chỉnh sửa báo cáo đợt 2 * Viết và code mô phỏng 1 tình huống tranh chấp lỗi đồng thời | **100%** |
| **3** | 18127055 | Hoàng Nguyên Trúc | * Viết và code mô phỏng 3 tình huống tranh chấp lỗi đồng thời | **100%** |

Mục lục

[THÔNG TIN NHÓM 2](#_Toc60433253)

[NGUỒN THAM KHẢO 4](#_Toc60433254)

[CÀI ĐẶT TÌNH HUỐNG TRANH CHẤP 5](#_Toc60433255)

[I. Sinh viên thực hiện: Phạm Ngọc Thùy Trang 5](#_Toc60433256)

[1. Tình huống 1: 5](#_Toc60433257)

[2. Tình huống 2: 13](#_Toc60433258)

[3. Tình huống 3: 18](#_Toc60433259)

[II. Sinh viên thực hiện: Hoàng Nguyên Trúc 23](#_Toc60433260)

[4. Tình huống 4: 23](#_Toc60433261)

[5. Tình huống 5: 32](#_Toc60433262)

[6. Tình huống 6: 34](#_Toc60433263)

[III. Sinh viên thực hiện: Ngô Thị Thanh Thảo 36](#_Toc60433264)

[7. Tình huống 7: 36](#_Toc60433265)

# NGUỒN THAM KHẢO

[1]: <https://stackoverflow.com/questions/2065627/repeatable-read-am-i-understanding-this-right>

[2]: Slide lý thuyết của cô Phạm Thị Bạch Huệ

[3]: Script code demo của cô Hồ Thị Hoàng Vy

[4]: <https://stackoverflow.com/questions/2065627/repeatable-read-am-i-understanding-this-right>

# CÀI ĐẶT TÌNH HUỐNG TRANH CHẤP

# Sinh viên thực hiện: Phạm Ngọc Thùy Trang

## Tình huống 1:

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: *Phantom***  T1 (User = NHANVIEN): Thống kê những nhà cần bán/cho thuê ở một địa điểm nhất định  T2 (User = CHUNHA): Thêm một nhà | | | |
| Thống kê những nhà cần bán/cho thuê ở một địa điểm nhất định | **Khóa** | Thêm một nhà | **Khóa** |
| ***Input:*** @MANV CHAR(5) = 'NV001'  ,@QUAN NVARCHAR(30) = NULL, @TPHO NVARCHAR(40) = 'TP.HCM' . Người dùng có thể dùng input mặc định hoặc thay đổi nếu muốn  ***Output***:  --Danh sách các nhà cần bán/cho thuê trong 3 tháng trở lại đây  -- Bài thống kê danh sách các nhà được đăng bán/cho thuê ở chi nhánh của nhân viên đó làm việc trong 3 tháng gồm có tên nhân viên lập thống kê, ngày báo cáo, danh sách, số lượng nhà đã bán, đã cho thuê, và tổng số lượng nhà chưa cho bán, cho thuê | ***Input***: @LOAINHA CHAR(5),  @NGAYHETHAN DATE,  @SLPHONG INTEGER,  @TINHTRANG NVARCHAR(100),  @GIABAN INTEGER,  @KHUVUC NVARCHAR(30),  @DUONG NVARCHAR(30),  @QUAN NVARCHAR(30),  @TPHO NVARCHAR(40),  @CHINHANHDK CHAR(5),  @DIEUKIENCHUNHA NVARCHAR(50)  ***Output***: Nếu nhà mới được thêm thành công thì  'Thêm nhà bán mới thành công',  Ngược lại nếu có lỗi xảy ra thì in ra lỗi tương ứng |
| SET TRANSACTION ISOLATION LEVEL REPEATABLE READ | SET TRANSACTION ISOLATION LEVEL READ COMMITTED |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra thông tin nhân viên có tồn tại hay không, nếu không tồn tại thì in câu lệnh báo lỗi và rollback transaction  IF (NOT EXISTS (SELECT \* FROM NHANVIEN WHERE NHANVIEN.MA\_NV = @MANV))  BEGIN  PRINT N'Không tồn tại mã nhân viên này'  ROLLBACK TRANSACTION;  RETURN;  END | **R(A)**  T1 xin khóa đọc trên bảng NHANVIEN với điều kiện Ma\_NV = @MANV và giữ khóa cho đến hết giao tác T1. T2 muốn cập nhật trên đơn vị dữ liệu này phải đợi T1 thực hiện xong |  |  |
| B2: Tìm ra chi nhánh mà nhân viên có @MANV đó đang làm việc  DECLARE @CHINHANH\_LAMVIEC CHAR(5) = (SELECT NHANVIEN.LAMVIEC FROM NHANVIEN WHERE NHANVIEN.MA\_NV = @MANV) | **R(A)** |  |  |
| B3: In danh sách các nhà cần thuê trong 3 tháng trở lại đây  IF(@QUAN IS NULL AND @TPHO IS NOT NULL)  BEGIN  SELECT N.MA\_NHA, N.SLPHONG, N.VIEW\_NHA, N.NVPT, N.MA\_CN, N.MA\_CNHA, LN.TENLN, N.NGAYDANG, N.TRANGTHAI  FROM NHA AS N, LOAINHA AS LN, DCHINHA AS DC  WHERE N.MA\_CN = @CHINHANH\_LAMVIEC AND LN.MA\_LN = N.MALN AND DC.MA\_NHA = N.MA\_NHA AND DC.TPHO = @TPHO  END | **R(B1, B, B2)**  T1 xin khóa đọc trên bảng NHA, LOAINHA, DCHINHA với điều kiện N.MA\_CN = @CHINHANH\_LAMVIEC AND LN.MA\_LN = N.MALN AND DC.MA\_NHA = N.MA\_NHA AND DC.TPHO = @TPHO và giữ khóa cho đến hết giao tác T1. T2 muốn cập nhật trên đơn vị dữ liệu này phải đợi T1 thực hiện xong |  |  |
| B4: -- In thông tin nhân viên làm báo cáo  SELECT NHANVIEN.TEN\_NV, @NGAYBAOCAO, NHANVIEN.LAMVIEC  FROM NHANVIEN WHERE NHANVIEN.MA\_NV = @MANV; | **R(A)** |  |  |
| WAITFOR DELAY '00:00:06' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: Tạo ra mã nhà mới bằng hàm và kiểm tra thông tin loại nhà insert vào có tồn tại hay không?  IF NOT EXISTS (SELECT \* FROM LOAINHA WHERE LOAINHA.MA\_LN = @LOAINHA)  BEGIN  PRINT N'Loại nhà này không tồn tại'  ROLLBACK TRANSACTION  RETURN;  END | **R(C)**  T2 xin khóa đọc trên bảng  LOAINHA  với điều kiện MA\_LN = @LOAINHA và nhả khóa sau khi kiểm tra xong. |
|  |  | B2: Kiểm tra chi nhánh đăng insert có tồn tại hay không?  IF NOT EXISTS (SELECT \* FROM CHINHANH WHERE CHINHANH.MA\_CN = @CHINHANHDK)  BEGIN  PRINT N'Chi nhánh này không tồn tại'  ROLLBACK TRANSACTION  RETURN;  END | **R(D)**  T2 xin khóa đọc trên bảng  CHINHANH  với điều kiện CHINHANH.MA\_CN = @CHINHANHDK và nhả khóa sau khi kiểm tra xong. |
|  |  | B3: Mặc định lấy mã nhân viên đầu tiên có làm việc tại chi nhánh được truyền vào. Set giá trị cho biến nhân viên phụ trách nhà  DECLARE @NVPHUTRACH CHAR(5) = (SELECT TOP 1 NHANVIEN.MA\_NV FROM NHANVIEN  WHERE NHANVIEN.LAMVIEC = @CHINHANHDK  ORDER BY NHANVIEN.MA\_NV ASC) | **R(E)**  T2 xin khóa đọc trên bảng  NHANVIEN  với điều kiện NHANVIEN.LAMVIEC = @CHINHANHDK và nhả khóa sau khi kiểm tra xong. |
|  |  | B4: Thêm dữ liệu mới vào bảng NHA, NHABAN, DCHINHA  INSERT INTO NHA(MA\_NHA,NGAYDANG,  NGAYHETHAN,  SLPHONG,TINHTRANG,  VIEW\_NHA,NVPT,MA\_CN,  MALN,MA\_CNHA,TRANGTHAI)  INSERT INTO NHABAN VALUES(@MANHAMOI, @GIABAN, @DIEUKIENCHUNHA);  INSERT INTO DCHINHA VALUES(@MANHAMOI, @KHUVUC, @DUONG, @QUAN, @TPHO);  PRINT N'Thêm nhà bán mới thành công' | **W(B1’, F, B2’)**  T2 xin khóa ghi trên các bảng NHA (B1’), NHABAN (F), DCHINHA  Cho đến khi commit  . |
|  |  | COMMIT |  |
| B5: -- Tiếp tục in danh sách các nhà lại một lần nữa  SELECT N.MA\_NHA, N.SLPHONG, N.VIEW\_NHA, N.NVPT, N.MA\_CN, N.MA\_CNHA, LN.TENLN, N.NGAYDANG, N.TRANGTHAI  FROM NHA AS N, LOAINHA AS LN, DCHINHA AS DC  WHERE N.MA\_CN = @CHINHANH\_LAMVIEC AND LN.MA\_LN = N.MALN AND DC.MA\_NHA = N.MA\_NHA AND DC.TPHO = @TPHO | **R(B1, B, B2, B1’, B2’)**  Lúc này, T1 đã phải đọc thêm dữ liệu vừa mới được ghi ở bên T2 cho nên hiển thị kết quả ra khác ban đầu |  |  |
| B6: -- In tổng số lượng nhà đã cho bán, đã cho thuê  SELECT COUNT(\*) AS SUMNHADABAN\_THUE  FROM NHA, DCHINHA  WHERE NHA.MA\_CN = @CHINHANH\_LAMVIEC AND DCHINHA.MA\_NHA = NHA.MA\_NHA AND DCHINHA.TPHO = @TPHO AND NHA.TRANGTHAI = 1  GROUP BY NHA.TRANGTHAI | **R(B1, B2)**  T1 xin khóa đọc trên bảng NHA, DCHINHA với điều kiện NHA.MA\_CN = @CHINHANH\_LAMVIEC AND DCHINHA.MA\_NHA = NHA.MA\_NHA AND DCHINHA.TPHO = @TPHO AND NHA.TRANGTHAI = 1 và giữ khóa cho đến hết giao tác T1. |  |  |
| B7: -- In tổng số lượng nhà chưa cho bán, chưa cho thuê  SELECT COUNT(\*) AS SUMNHACHUABAN\_THUE  FROM NHA, DCHINHA  WHERE NHA.MA\_CN = @CHINHANH\_LAMVIEC AND DCHINHA.MA\_NHA = NHA.MA\_NHA AND DCHINHA.TPHO = @TPHO AND NHA.TRANGTHAI = 0  GROUP BY NHA.TRANGTHAI | **R(B1, B1’, B2, B2’)**  T1 xin khóa đọc trên bảng NHA, DCHINHA với điều kiện NHA.MA\_CN = @CHINHANH\_LAMVIEC AND DCHINHA.MA\_NHA = NHA.MA\_NHA AND DCHINHA.TPHO = @TPHO AND NHA.TRANGTHAI = 0 và giữ khóa cho đến hết giao tác T1.  Lúc này, T1 đã phải đọc thêm dữ liệu vừa mới được ghi ở bên T2 cho nên hiển thị kết quả ra khác ban đầu  . |  |  |
| COMMIT |  |  |  |
|  |  |  |  |

Ta có thể hình dung đơn giản đoạn gây lỗi như sau:

A picture containing graphical user interface

Description automatically generated

## Tình huống 2:

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: *Unrepeatable data***  T1 (User = KHACHHANG): Xem thông tin chi tiết nhà bán  T2 (User = CHUNHA): Cập nhật thông tin, chỉnh sửa của một nhà bán | | | |
| Xem thông tin chi tiết nhà bán | **Khóa** | Cập nhật thông tin, chỉnh sửa của một nhà bán | **Khóa** |
| ***Input:*** @MANHA CHAR(5), @MUAHAYKHONG BIT = 0  ***Output***: Thông tin chi tiết của nhà, bình luận về nhà, nếu @MUAHAYKHONG = 1 thì sẽ hiển thị thêm thông tin sơ lược về nhà đăng ký mua.  In ra dòng chữ “N'CHỨC NĂNG NÀY ĐÃ THỰC HIỆN THÀNH CÔNG' nếu chức năng không gặp phải bất cứ lỗi nào | ***Input***: @MANHA CHAR(5) = NULL,@MA\_LN CHAR(5) = NULL, @GIABAN INTEGER = NULL, @SLP INTEGER = NULL, @KHUVUC NVARCHAR(30) = NULL, @DUONG NVARCHAR(30) = NULL, @QUAN NVARCHAR(30) = NULL,@TPHO NVARCHAR(40) = NULL, @NHH DATE = NULL, @TINHTRANG NVARCHAR(100) = NULL, @CHINHANH CHAR(5) = NULL  ***Output***: Nhà bán sau khi đã được cập nhật, chỉnh sửa thông tin |
| SET TRANSACTION ISOLATION  LEVEL READ COMMITTED | SET TRANSACTION ISOLATION  LEVEL READ COMMITTED |
| BEGIN TRAN |  |  |  |
| B1: Khai báo các biến như :  @TENCNHA, @SLPHONG, @LOAINHA, @GIABAN, @TINHTRANG, @DKCHUNHA, @TRANGTHAI, @XEM, @NGAYDANGBAI, @NGAYHETHAN, @DIACHI, @MAKH, @BINHLUAN |  |  |  |
| B2: Set giá trị cho các biến, cụ thể như:  SET @DIACHI=(SELECT DCHINHA.DUONG FROM DCHINHA WHERE DCHINHA.MA\_NHA=@MANHA)+' '+(SELECT DCHINHA.QUAN FROM DCHINHA WHERE DCHINHA.MA\_NHA=@MANHA)+' '+(SELECT DCHINHA.TPHO FROM DCHINHA WHERE DCHINHA.MA\_NHA=@MANHA) | **R(A)**  T1 xin cấp khóa đọc cho bảng DCHINHA với điều kiện MA\_NHA = @MANHA.  Khóa sẽ được trả sau khi  quá trình đọc hoàn tất |  |  |
| SET @LOAINHA=(SELECT LOAINHA.TENLN FROM LOAINHA,NHA WHERE NHA.MA\_NHA=@MANHA AND LOAINHA.MA\_LN=NHA.MALN) | **R(B)**  T1 xin cấp khóa đọc cho bảng LOAINHA với điều kiện NHA.MA\_NHA=  @MANHA  =@MANHA AND LOAINHA.MA\_LN  =NHA.MALN. Khóa sẽ được trả sau khi quá trình đọc hoàn tất |  |  |
| SET @GIABAN=(SELECT NHABAN.GIABAN FROM NHABAN WHERE NHABAN.MA\_NHA=@MANHA) | **R(C)**  T1 xin cấp khóa đọc cho bảng NHABAN với điều kiện NHABAN.MA\_NHA=  @MANHA Khóa sẽ được trả sau khi quá trình đọc hoàn tất |  |  |
| …. Tương tự như vậy với quá trình set giá trị cho các biến khác |  |  |  |
| WAITFOR DELAY '00:00:06' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: Kiểm tra xem nếu trường nhập vào không có giá trị NULL thì update giá trị.  Chẳng hạn, kiểm tra nếu @MA\_LN không bằng  NULL thì update lại mã loại nhà  IF @MA\_LN IS NOT NULL  BEGIN  UPDATE NHA SET MALN = @MA\_LN  WHERE MA\_NHA = @MANHA  END | **W(B’)**  T2 xin cấp khóa ghi cho bảng NHA với điều kiện MA\_NHA = @MANHA. Khóa sẽ được trả sau khi T2 commit |
|  |  | IF @GIABAN IS NOT NULL  BEGIN  UPDATE NHABAN SET GIABAN = @GIABAN WHERE  MA\_NHA = @MANHA  END | **W(C’)**  T2 xin cấp khóa ghi cho bảng NHABAN với điều kiện MA\_NHA = @MANHA. Khóa sẽ được trả sau khi T2 commit |
|  |  | …. Tương tự như vậy với quá trình kiểm tra và update giá trị cho các biến khác |  |
|  |  | COMMIT |  |
| B3 : Kiểm tra xem nếu @MUAHAYKHONG = 1 thì ta kiểm tra tiếp trạng thái nhà bán lại một lần nữa xem nhà đã bán hay chưa. Nếu chưa thì hiển thị thông tin sơ lược về nhà đăng ký mua  PRINT N'THÔNG TIN NHÀ BẠN ĐĂNG KÝ MUA SƠ LƯỢC GỒM'  SELECT NHA.MA\_NHA, CHUNHA.TEN\_CNHA, NHABAN.GIABAN  FROM NHA, CHUNHA, NHABAN WHERE NHABAN.MA\_NHA = @MANHA AND NHA.MA\_NHA = @MANHA AND CHUNHA.MA\_CNHA = NHA.MA\_CNHA  PRINT N'CHỨC NĂNG NÀY ĐÃ THỰC HIỆN THÀNH CÔNG' | **R(B’, C’)**  T1 xin cấp khóa đọc cho bảng NHABAN, NHA, CHUNHA với điều kiện  NHABAN.MA\_NHA = @MANHA AND NHA.MA\_NHA = @MANHA AND CHUNHA.MA\_CNHA = NHA.MA\_CNHA. Khóa sẽ được trả sau khi quá trình đọc hoàn tất  T1 đã đọc giá trị B’ (là giá trị cập nhật của B bởi T2) và giá trị C’ (là giá trị cập nhật của C bởi T2) nên đã xảy ra tình trạng đọc cùng một dòng dữ liệu nhưng lại cho ra 2 kết quả khác nhau |  |  |
| COMMIT |  |  |  |

Ta có thể hình dung đơn giản đoạn gây lỗi như sau:

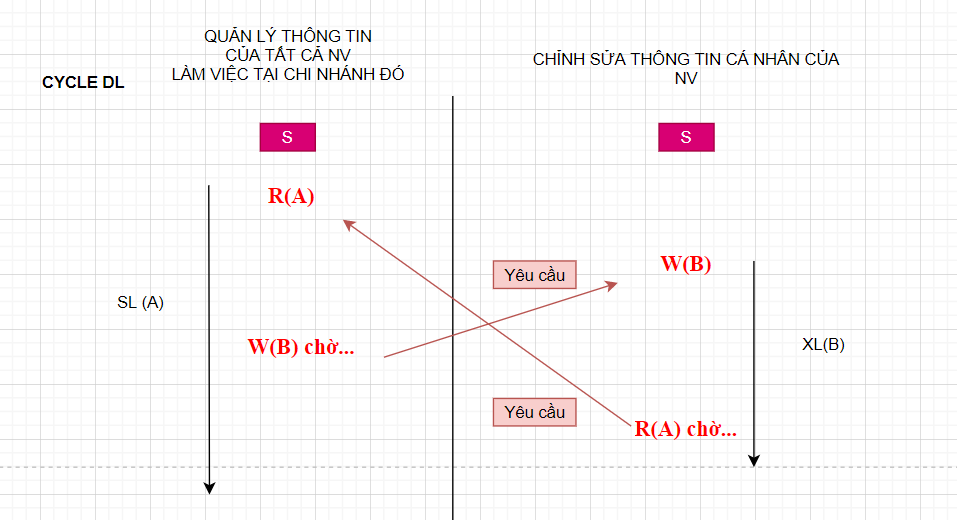
Graphical user interface, application, Teams

Description automatically generated

## Tình huống 3:

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: *Cycle Deadlock***  T1 (User = NHANVIENQUANLY): Quản lý thông tin của tất cả nhân viên làm việc tại chi nhánh đó  T2 (User = NHANVIEN): Chỉnh sửa thông tin cá nhân | | | |
| Quản lý thông tin của tất cả nhân viên làm việc tại chi nhánh đó | **Khóa** | Chỉnh sửa thông tin cá nhân | **Khóa** |
| ***Input:*** @NVQL CHAR(5),  @MANV CHAR(5) = NULL, /\*Gía trị mặc định là null\*/  @UPDATESALARY BIT = 0,  @LUONG FLOAT = 0  ***Output***: Thông tin của tất cả các nhân viên làm việc tại chi nhánh đó và lương của 1 nhân viên bất kỳ được cập nhật nếu có | ***Input***: @MANV CHAR(5), @TENNV NVARCHAR(50) = NULL, @SDT VARCHAR(15) = NULL, @DCHI NVARCHAR(250) = NULL, @DOB DATE = NULL  ***Output***: Thông tin đã được chỉnh sửa và dòng thông báo chúc mừng thành công hoặc thông báo lỗi nếu có lỗi |
| SET TRANSACTION ISOLATION LEVEL SERIALIZABLE | SET TRANSACTION ISOLATION LEVEL SERIALIZABLE |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra thông tin xem và không cho phép mã nhân viên truyền vào là 1 nhân viên quản lý khác  IF (EXISTS (SELECT \* FROM CHINHANH WHERE CHINHANH.NVQUANLY = @MANV))  BEGIN  PRINT N'Không được xóa/cập nhật  lương của nhân viên quản lý khác';  ROLLBACK TRANSACTION  RETURN;  END | **R(X)**  Xin khóa đọc trên bảng CHINHANH với điều kiện  CHINHANH.NVQUANLY = @MANV và chỉ trả sau khi commit. T2 muốn xin khóa đọc trên đơn vị dữ liệu này thì phải đợi T1 nhả khóa |  |  |
| B2: Tìm chi nhánh của nhân viên quản lý được truyền vào  DECLARE @CHINHANHNVQL CHAR(5) = (SELECT NHANVIEN.LAMVIEC FROM NHANVIEN WHERE NHANVIEN.MA\_NV = @NVQL) | **R(E)**  Xin khóa đọc trên bảng NHANVIEN với điều kiện  NHANVIEN.MA\_NV = @NVQL và giữ khóa cho tới khi commit. T2 muốn xin khóa đọc trên đơn vị dữ liệu này thì phải đợi T1 nhả khóa |  |  |
| B3: Hiển thị danh sách các nhân viên do nhân viên quản lý đó quản lý khi **chưa** cập nhật lương  SELECT NHANVIEN.MA\_NV, NHANVIEN.TEN\_NV,  NHANVIEN.LUONG, NHANVIEN.LAMVIEC FROM NHANVIEN WHERE NHANVIEN.LAMVIEC = @CHINHANHNVQL | **R(A)**  Xin khóa đọc trên bảng NHANVIEN với điều kiện  NHANVIEN.LAMVIEC = @CHINHANHNVQL và giữ khóa cho tới khi commit . T2 muốn xin khóa đọc trên đơn vị dữ liệu này thì phải đợi T1 nhả khóa |  |  |
| B4: Khi nhân viên đó muốn cập nhật lương thì kiểm tra nhân viên đó có tồn tại trong bảng nhân viên hay không?  IF(NOT EXISTS (SELECT \* FROM NHANVIEN WHERE NHANVIEN.MA\_NV = @MANV AND NHANVIEN.LAMVIEC = @CHINHANHNVQL))  BEGIN  PRINT N'Không tồn tại nhân  viên này trong bảng nhân viên'  ROLLBACK TRANSACTION  RETURN;  END | **R(D)**  Xin khóa đọc trên bảng NHANVIEN với điều kiện  NHANVIEN.MA\_NV = @MANV AND NHANVIEN.LAMVIEC = @CHINHANHNVQL và chỉ trả sau khi commit. T2 muốn xin khóa đọc trên đơn vị dữ liệu này thì phải đợi T1 nhả khóa |  |  |
| WAITFOR DELAY '00:00:07' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: Kiểm tra xem nhân viên truyền vào có tồn tại hay không  IF (NOT EXISTS (SELECT \* FROM NHANVIEN WHERE NHANVIEN.MA\_NV = @MANV))  BEGIN  PRINT 'Nhân viên này không tồn tại'  ROLLBACK TRANSACTION  RETURN;  END | **R(B)**  Xin khóa đọc trên bảng NHANVIEN với điều kiện  NHANVIEN.MA\_NV = @MANV và chỉ trả sau khi commit. T1 muốn xin khóa đọc trên đơn vị dữ liệu này thì phải đợi T2 nhả khóa |
|  |  | B2: Cập nhật giá trị nếu trường vào không bằng NULL, chẳng hạn như  IF @TENNV IS NOT NULL  UPDATE NHANVIEN  SET NHANVIEN.TEN\_NV = @TENNV  WHERE NHANVIEN.MA\_NV = @MANV | **W(B)**  Xin khóa ghi trên bảng NHANVIEN với điều kiện  NHANVIEN.MA\_NV = @MANV và chỉ trả sau khi T2 commit. T1 muốn xin khóa đọc trên đơn vị dữ liệu này thì phải đợi T2 nhả khóa |
|  |  | B3: Tìm chi nhánh quản lý của nhân viên được truyền vào  DECLARE @CHINHANHNVQL CHAR(5) = (SELECT NHANVIEN.LAMVIEC FROM NHANVIEN WHERE NHANVIEN.MA\_NV = @MANV) | **R(B)** |
|  |  | B4: In ra thông tin sau khi chỉnh sửa  SELECT NHANVIEN.MA\_NV, NHANVIEN.TEN\_NV,  NHANVIEN.LUONG, NHANVIEN.LAMVIEC FROM NHANVIEN WHERE NHANVIEN.LAMVIEC = @CHINHANHNVQL AND NHANVIEN.MA\_NV = @MANV | **R(A)**  Xin khóa ghi trên bảng NHANVIEN với điều kiện  NHANVIEN.LAMVIEC = @CHINHANHNVQL AND NHANVIEN.MA\_NV = @MANV  **Tuy nhiên xin khóa không được do T1 vẫn đang giữ khóa trên đơn vị dữ liệu này và vẫn không chịu nhả cho tới khi T1 commit** |
|  |  | COMMIT |  |
| B5: Cập nhật lương cho nhân viên được chỉ định  UPDATE NHANVIEN SET NHANVIEN.LUONG = @LUONG  WHERE NHANVIEN.MA\_NV = @MANV | **W(B)**  Xin khóa GHI trên bảng NHANVIEN với điều kiện  NHANVIEN.MA\_NV = @MANV  **Tuy nhiên xin khóa không được do T2 vẫn đang giữ khóa trên đơn vị dữ liệu này và vẫn không chịu nhả cho tới khi T2 commit** |  |  |
| B6: Hiển thị danh sách các nhân viên do nhân viên quản lý đó quản lý khi **đã** cập nhật lương  SELECT NHANVIEN.MA\_NV, NHANVIEN.TEN\_NV,  NHANVIEN.LUONG, NHANVIEN.LAMVIEC FROM NHANVIEN WHERE NHANVIEN.LAMVIEC = @CHINHANHNVQL | **R(A)** |  |  |
| COMMIT |  |  |  |

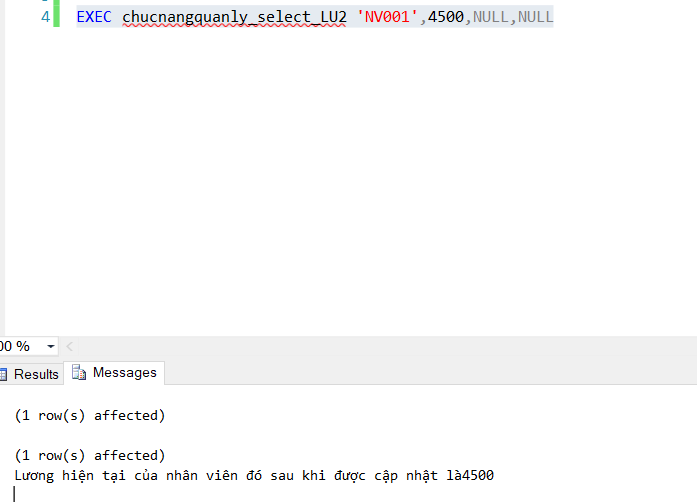
Ta có thể hình dung đơn giản đoạn gây lỗi như sau:

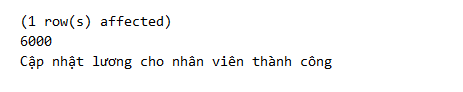


## Tình huống 8:

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: *Lost Update***  T1 (User = NHANVIENQUANLY): Tăng/giảm lương định kỳ cho nhân viên và bonus thêm tiền hoa hồng cho nhân viên đó nếu có  T2 (User = NHANVIEN): Quản lý thông tin của một nhân viên bất kỳ | | | |
| Tăng/giảm lương định kỳ cho nhân viên và bonus thêm tiền hoa hồng cho nhân viên đó nếu có | **Khóa** | Quản lý thông tin của một nhân viên bất kỳ | **Khóa** |
| ***Input:*** @NVQL CHAR(5),  @MANV CHAR(5) = NULL, /\*Gía trị mặc định là null\*/  @UPDATESALARY BIT = 0,  @LUONG FLOAT = 0  ***Output***: Lương nhân viên sau khi được cập nhật | ***Input***: @MANV CHAR(5), @LUONG FLOAT , @SDT VARCHAR(15), @DCHI\_NV NVARCHAR(250)  ***Output***: Thông tin của nhân viên (ở đây là lương) sau khi được cập nhật |
| SET TRANSACTION ISOLATION LEVEL READ COMMITTED | SET TRANSACTION ISOLATION LEVEL READ COMMITTED |
| BEGIN TRANSACTION |  |  |  |
| B1: Kiểm tra xem mã NV truyền vào có tồn tại hay không  IF (NOT EXISTS (SELECT \* FROM NHANVIEN WHERE NHANVIEN.MA\_NV = @MANV))  BEGIN  print N'Không tồn tại mã nhân viên này'  rollback transaction  return;  END | **R(NHANVIEN)**  T1 xin khóa đọc trên đơn vị dữ liệu là bảng NHANVIEN và trả khóa sau khi đã đọc xong |  |  |
| B2: Tìm số lượng nhà mà nhân viên đó phụ trách và tiền lương hiện tại của nhân viên đó  DECLARE @SONHAPHUTRACH INT = (SELECT COUNT(MA\_NHA)  FROM NHA  WHERE NVPT = @MANV  GROUP BY NVPT)  DECLARE @LUONGNV FLOAT  SELECT @LUONGNV = LUONG  FROM NHANVIEN WHERE NHANVIEN.MA\_NV = @MANV  DECLARE @SOTIENBITRU FLOAT = 0; | **R(NHA)**  T1 xin khóa đọc trên đơn vị dữ liệu là bảng NHA và trả khóa sau khi đã đọc xong |  |  |
| /\*Nếu số nhà phụ trách lớn hơn 10 ==> thưởng 5000 \*/  /\*Nếu số nhà phụ trách < 10 && >= 5 ==> trừ 2000 \*/  /\*Nếu số nhà phụ trách < 5 && >= 2 ==> trừ 3000 \*/  /\*Nếu số nhà phụ trách < 2 ==> trừ 4500 \*/  WAITFOR DELAY '00:00:07' |  |  |  |
|  |  | BEGIN TRANSACTION |  |
|  |  | B1: In ra thông tin hiện tại của nhân viên có mã được truyền vào  select MA\_NV, TEN\_NV, SDT, DCHI\_NV, NGSINH, LUONG, USER\_NV, LAMVIEC  from NHANVIEN  where MA\_NV = @MANV | **R(NHANVIEN)**  T2 xin khóa đọc trên đơn vị dữ liệu là bảng NHANVIEN và trả khóa sau khi đã đọc xong |
|  |  | B2: Kiểm tra xem các thông số truyền vào có bằng NULL hay không, nếu không thì cập nhật giá trị.  IF @LUONG IS NOT NULL  BEGIN  UPDATE NHANVIEN SET LUONG = @LUONG WHERE MA\_NV = @MANV | **W(NHANVIEN)**  T2 xin khóa ghi trên đơn vị dữ liệu là bảng NHANVIEN và trả khóa sau khi T2 commit |
|  |  | B3: Kiểm tra xem các thông số truyền vào có bằng NULL hay không, nếu không thì cập nhật giá trị.  DECLARE @LUONGHIENTAI FLOAT  SELECT @LUONGHIENTAI = (SELECT LUONG FROM NHANVIEN WHERE NHANVIEN.MA\_NV = @MANV)  PRINT N'Lương hiện tại của nhân viên đó sau khi được cập nhật là' + CONVERT(VARCHAR, @LUONGHIENTAI)  END | **R(NHANVIEN)** |
|  |  | commit transaction |  |
| B4: Kiểm tra xem tương ứng với số nhà phụ trách sẽ được tăng/giảm lương như thế nào cũng như cộng thêm tiền hoa hồng nếu có  IF(@SONHAPHUTRACH >= 10)  BEGIN  SET @SOTIENBITRU = 5000  SET @LUONGNV = @LUONGNV + @SOTIENBITRU + @BONUSTIP  ---Cập nhật lương  UPDATE NHANVIEN SET LUONG = @LUONGNV  WHERE NHANVIEN.MA\_NV = @MANV  END  ELSE IF (@SONHAPHUTRACH >=5 AND @SONHAPHUTRACH <10 )  BEGIN  SET @SOTIENBITRU = 2000  SET @LUONGNV = @LUONGNV + @BONUSTIP - @SOTIENBITRU  ---Cập nhật lương  UPDATE NHANVIEN SET LUONG = @LUONGNV  WHERE NHANVIEN.MA\_NV = @MANV  END  ELSE IF (@SONHAPHUTRACH >=2 AND @SONHAPHUTRACH < 5)  BEGIN  SET @SOTIENBITRU = 3000  SET @LUONGNV = @LUONGNV + @BONUSTIP - @SOTIENBITRU  ---Cập nhật lương  UPDATE NHANVIEN SET LUONG = @LUONGNV  WHERE NHANVIEN.MA\_NV = @MANV  END  ELSE IF (@SONHAPHUTRACH < 2)  BEGIN  SET @SOTIENBITRU = 4500  SET @LUONGNV = @LUONGNV + @BONUSTIP - @SOTIENBITRU  ---Cập nhật lương  UPDATE NHANVIEN SET LUONG = @LUONGNV  WHERE NHANVIEN.MA\_NV = @MANV  END  PRINT @LUONGNV | **W(NHANVIEN)**  T1 xin khóa ghi trên đơn vị dữ liệu là bảng NHANVIEN và trả khóa sau khi T1 commit |  |  |
| commit transaction |  |  |  |

Giả sử ban đầu NV001 có lương là 4000. Ta mong đợi rằng sau khi được cập nhật qua cả 2 T thì lương của nhân viên sẽ là 4500 *(do T2 sẽ commit trước T1)* + (5000 – 3000) = 6500

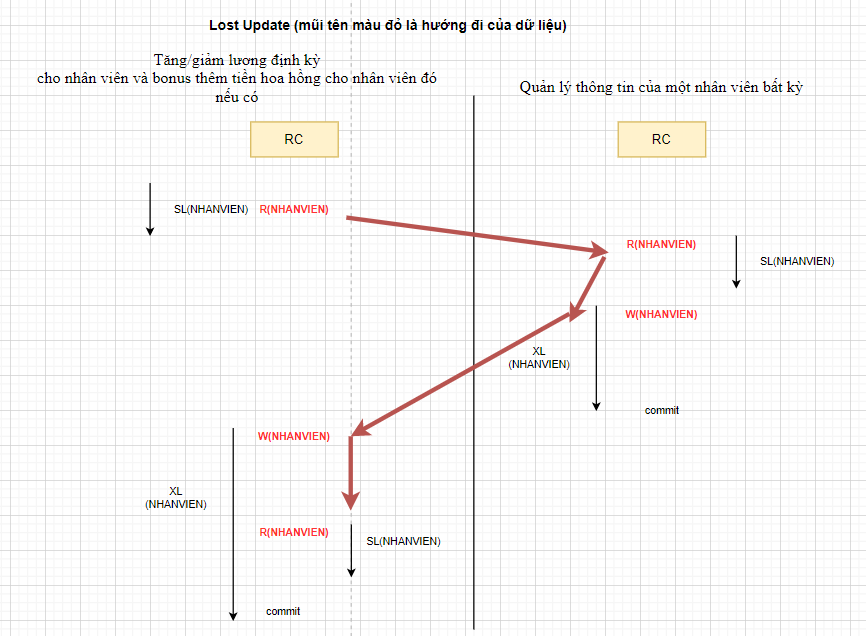




**Ta để ý rằng T2 commit trước T1. Và in ra kết quả lương hiện tại của nhân viên đó (cụ thể ở đây là nhân viên có mã nhân viên là ‘NV001’) là** 4500**. Sau đó 1 khoảng thời gian thì T1 cũng hoàn tất việc thực thi và in ra giá trị lương hiện tại của nhân viên đó sau khi được cập nhật là** 6000

**Cả 2 giá trị trên đều sai vì thực chất giá trị ta đang mong đợi phải là 6500 chứ không phải là 6000. Như vậy bằng một cách thần kỳ nào đó mà 500đ của nhân viên ‘NV001’ đã bị mất khi có cả 2T cùng thực hiện chức năng cập nhật LƯƠNG cho nhân viên này 🡺 lỗi tranh chấp “Lost Update”. *Tình trạng này xảy ra khi có nhiều hơn một giao tác cùng thực hiện cập nhật trên 1 đơn vị dữ liệu. Khi đó, tác dụng của giao tác cập nhật thực hiện sau sẽ đè lên tác dụng của thao tác cập nhật trước.***

Ta có thể hình dung đơn giản đoạn gây lỗi như sau:

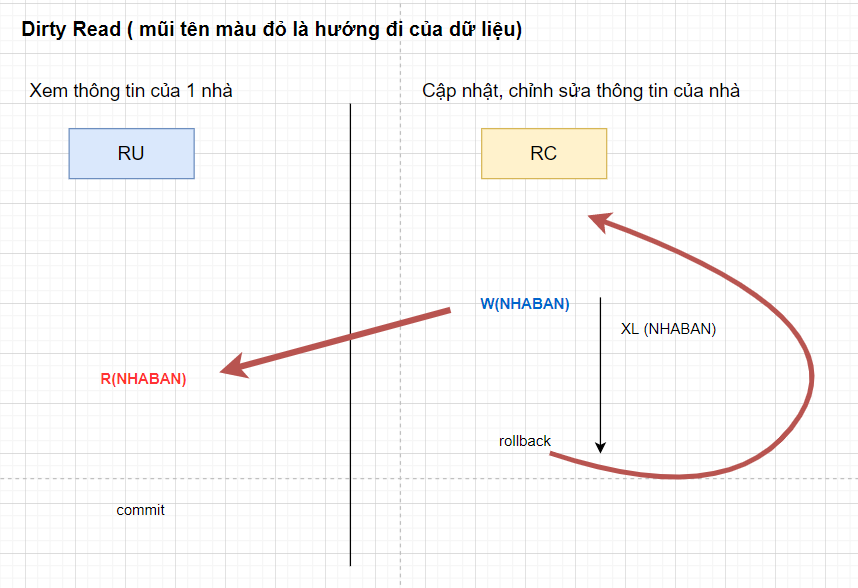


# Sinh viên thực hiện: Hoàng Nguyên Trúc

## Tình huống 4:

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: *Dirty Read***  T1 (User = KHACHHANG): Xem thông tin chi tiết của một nhà  T2 (User = CHUNHA): Cập nhật, chỉnh sửa thông tin của nhà | | | |
| thongtinchitiet\_NB | **Khóa** | chinhsua\_loainha\_ban | **Khóa** |
| ***Input:*** @MANHA CHAR(5), @MUAHAYKHONG BIT = 0  ***Output***: Hiện ra thông tin chi tiết của nhà bán | ***Input***: @MANHA CHAR(5) = NULL,@MA\_LN CHAR(5) = NULL, @GIABAN INTEGER = NULL, @SLP INTEGER = NULL,  @KHUVUC NVARCHAR(30) = NULL, @DUONG NVARCHAR(30) = NULL, @QUAN NVARCHAR(30) = NULL,  @TPHO NVARCHAR(40) = NULL, @NHH DATE = NULL, @TINHTRANG NVARCHAR(100) = NULL, @CHINHANH CHAR(5) = NULL  ***Output***: Thông tin nhà bán được chỉnh sửa |
| SET TRANSACTION ISOLATION  LEVEL **READ UNCOMMITTED** | SET TRANSACTION ISOLATION  LEVEL **READ COMMITTED** |
|  |  | BEGIN TRAN |  |
|  |  | **B1: Kiểm tra thông số truyền vào có NULL hay không, nếu không thì cập nhật giá trị**  **Chẳng hạn ở đây ta kiểm tra @MA\_LN có NULL hay không, nếu không thì cập nhật mã loại nhà**  IF @MA\_LN IS NOT NULL  BEGIN  UPDATE NHA SET MALN = @MA\_LN WHERE MA\_NHA = @MANHA  END | **W(NHABAN)**  Xin khóa ghi trên bảng NHÀ . T2 sẽ giữ khóa ghi này cho đến khi T2 commit |
|  |  | **B2: hẳng hạn ở đây ta kiểm tra @TINHTRANG có NULL hay không, nếu không thì cập nhật mã loại nhà**  IF @TINHTRANG IS NOT NULL  BEGIN  UPDATE NHA SET TINHTRANG = @TINHTRANG WHERE MA\_NHA = @MANHA  END | **W(NHABAN)** |
|  |  | **B3: Chẳng hạn ở đây ta kiểm tra @GIABAN có NULL hay không, nếu không thì cập nhật mã loại nhà. Tuy nhiên, nếu @GIABAN là một số không chia hết cho 100 thì rollback transaction**  IF @GIABAN IS NOT NULL  BEGIN  UPDATE NHABAN SET GIABAN = @GIABAN WHERE MA\_NHA = @MANHA  IF @GIABAN % 100 != 0 | **W(NHABAN)** |
|  |  | WAITFOR DELAY '00:00:10' |  |
| BEGIN TRAN |  |  |  |
| **B1: Khai báo các biến cần thiết**  DECLARE @TENCNHA NVARCHAR(20)  DECLARE @SLPHONG INTEGER  DECLARE @LOAINHA NVARCHAR(20)  DECLARE @GIABAN INTEGER  DECLARE @TINHTRANG NVARCHAR(100)  DECLARE @DKCHUNHA NVARCHAR(50)  DECLARE @TRANGTHAI BIT  DECLARE @XEM INTEGER  DECLARE @NGAYDANGBAI DATE  DECLARE @NGAYHETHAN DATE  DECLARE @DIACHI NVARCHAR(54) |  |  |  |
| **B2: Set giá trị cho các biến cần thiết**  SET @DIACHI=(SELECT DCHINHA.DUONG FROM DCHINHA WHERE DCHINHA.MA\_NHA=@MANHA)+' '+(SELECT DCHINHA.QUAN FROM DCHINHA WHERE DCHINHA.MA\_NHA=@MANHA)+' '+(SELECT DCHINHA.TPHO FROM DCHINHA WHERE DCHINHA.MA\_NHA=@MANHA)  SET @TENCNHA=(SELECT CHUNHA.TEN\_CNHA FROM CHUNHA,NHA WHERE NHA.MA\_NHA=@MANHA AND CHUNHA.MA\_CNHA=NHA.MA\_CNHA)  SET @SLPHONG=(SELECT NHA.SLPHONG FROM NHA WHERE NHA.MA\_NHA=@MANHA)  SET @LOAINHA=(SELECT LOAINHA.TENLN FROM LOAINHA,NHA WHERE NHA.MA\_NHA=@MANHA AND LOAINHA.MA\_LN=NHA.MALN)  SET @GIABAN=(SELECT NHABAN.GIABAN FROM NHABAN WHERE NHABAN.MA\_NHA=@MANHA)  SET @TINHTRANG=(SELECT NHA.TINHTRANG FROM NHA WHERE NHA.MA\_NHA=@MANHA)  SET @DKCHUNHA=(SELECT NHABAN.DKCHUNHA FROM NHABAN WHERE NHABAN.MA\_NHA=@MANHA)  SET @TRANGTHAI=(SELECT NHA.TRANGTHAI FROM NHA WHERE NHA.MA\_NHA=@MANHA)  SET @XEM=(SELECT NHA.VIEW\_NHA FROM NHA WHERE NHA.MA\_NHA=@MANHA)  SET @NGAYDANGBAI=(SELECT NHA.NGAYDANG FROM NHA WHERE NHA.MA\_NHA=@MANHA)  SET @NGAYHETHAN=(SELECT NHA.NGAYHETHAN FROM NHA WHERE NHA.MA\_NHA=@MANHA) | **R(NHABAN)**  **R(DCHINHA)**  **R(CHUNHA)**  **R(NHA)**  **R(LOAINHA)**  **Vì isolation level của T1 là Read Uncommitted nên các tiến trình đọc không cần đợi đến khi tiến trình ghi hoàn tất mà có thể lấy dữ liệu ra được ngay *(không cần cấp khóa đọc).* Chính vì lẽ đó mà giá trị cho biến @GIABAN đã có giá trị = giá bán do T2 bên kia cập nhật** |  |  |
| **B3: In ra màn hình các thông tin cần thiết**  PRINT(N'Xem :'+CAST(@XEM AS CHAR(3)))  PRINT(N'Ngày Đăng Bài: '+cast ( @NGAYDANGBAI as char(20)))  PRINT(N'Ngày Hết Hạn: '+ cast( @NGAYHETHAN as char(20)))  PRINT(N'Địa chỉ: '+@DIACHI)  PRINT(N'Chủ nhà: '+@TENCNHA)  PRINT(N'Số lượng phòng ở: '+ CAST( @SLPHONG as varchar(10)))  PRINT(N'Loại nhà:'+@LOAINHA)  PRINT(N'Giá bán: '+CAST(@GIABAN AS VARCHAR(10)))  PRINT(N'Tình trạng: '+@TINHTRANG)  PRINT(N'Điều kiện chủ nhà: '+@DKCHUNHA) | **Lúc này giá trị in ra là giá trị mà T2 vừa mới cập nhật** |  |  |
| **B4 : Kiểm tra nếu nhà đã mua rồi thì báo lỗi, ngược lại in thông tin chi tiết nhà sẽ mua**  IF (@MUAHAYKHONG = 1)  BEGIN  DECLARE @TRANGTHAINHA BIT = (SELECT NHA.TRANGTHAI FROM NHA WHERE NHA.MA\_NHA = @MANHA)  IF (@TRANGTHAINHA = 1)  BEGIN  PRINT N' NHÀ ĐÃ ĐƯỢC BÁN RỒI. KHÔNG THỂ MUA NỮA';  ROLLBACK TRANSACTION  RETURN;  END  ELSE BEGIN  PRINT N'THÔNG TIN NHÀ BẠN ĐĂNG KÝ MUA SƠ LƯỢC GỒM'  SELECT NHA.MA\_NHA, CHUNHA.TEN\_CNHA, NHABAN.GIABAN  FROM NHA, CHUNHA, NHABAN WHERE NHABAN.MA\_NHA = @MANHA AND NHA.MA\_NHA = @MANHA AND CHUNHA.MA\_CNHA = NHA.MA\_CNHA  END  END | **R(NHABAN)**  **R(NHA)**  **R(CHUNHA)**  **Tương tự như vậy giá trị in ra trên màn hình cũng là giá trị mà T2 vừa mới cập nhật** |  |  |
| COMMIT |  | rollback transaction  return; | **//Khi update dữ liệu GIABAN cho NHA, nhưng do GIABAN không thỏa điều kiện if nên điều này dẫn đến toàn dữ liệu sẽ bị ROLLBACK**  **//Như vậy toàn bộ dữ liệu mà T2 vừa mới cập nhật lúc này GIABAN đã không còn, T1 đã đọc lại giá trị cũ trước khi T2 rollback transaction** |

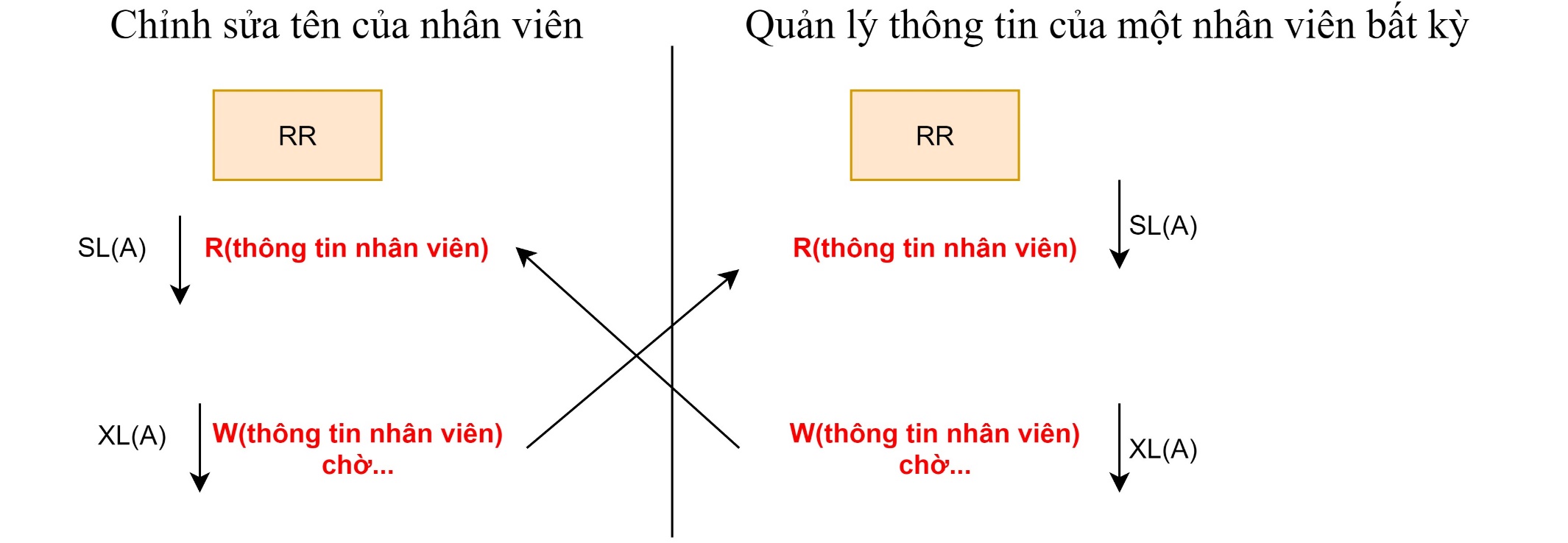
Ta có thể hình dung đơn giản đoạn gây lỗi như sau:



## Tình huống 5:

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: Conversion Deadlock**  T1 (User = NHANVIEN): Chỉnh sửa tên của nhân viên (chức năng nhỏ).  T2 (User = QUANLY): Quản lý thông tin của một nhân viên bất kỳ | | | |
| chucnangquanly\_select | **Khóa** | Chucnangquanly\_select\_nv | **Khóa** |
| ***Input:*** @MANV CHAR(5), @LUONG FLOAT , @SDT VARCHAR(15), @DCHI\_NV NVARCHAR(250)  ***Output***: Thông tin của nhân viên | ***Input***: @MANV CHAR(5) , @SDT VARCHAR(15), @DCHI\_NV NVARCHAR(250)  ***Output***: Thông tin của nhân viên |
| SET TRANSACTION ISOLATION  LEVEL **REPEATEABLE READ** | SET TRANSACTION ISOLATION  LEVEL **REPEATEABLE READ** |
| BEGIN TRAN |  |  |  |
| B1: Xem thông tin của nhân viên  select MA\_NV, TEN\_NV, SDT, DCHI\_NV, NGSINH, LUONG, USER\_NV, LAMVIEC  from NHANVIEN  where MA\_NV = @MANV | **R(NHANVIEN)**  Xin share lock để đọc trên bảng NHANVIEN với điều kiện NHANVIEN.MA\_NV= @MANV |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: Xem thông tin của nhân viên  select MA\_NV, TEN\_NV, SDT, DCHI\_NV, NGSINH, LUONG, USER\_NV, LAMVIEC  from NHANVIEN  where MA\_NV = @MANV | **R(NHANVIEN)**  Xin share lock để đọc trên bảng NHANVIEN với điều kiện NHANVIEN.MA\_NV= @MANV |
| WAITFOR DELAY '0:0:05' |  |  |  |
| B2: Cập nhật tên nhân viên  IF @SDT IS NOT NULL  BEGIN try  UPDATE NHANVIEN SET SDT = @SDT WHERE MA\_NV = @MANV  END try | **W(NHANVIEN)**  Xin khóa GHI trên bảng NHANVIEN với điều kiện NHANVIEN.MA\_NV= @MANV và chỉ trả sau khi commit |  |  |
|  |  |  |  |
|  |  | WAITFOR DELAY '0:0:05' |  |
|  |  | B2: Cập nhật tên nhân viên  begin try  UPDATE NHANVIEN SET SDT= @SDT WHERE MA\_NV = @MANV  end try | **W(NHANVIEN)**  Xin khóa ghi trên bảng NHANVIEN với điều kiện NHANVIEN.MA\_NV= @MANV |
|  |  | COMMIT |  |
| COMMIT |  |  |  |

Ta có thể hình dung đơn giản đoạn gây lỗi như sau:



## Tình huống 6:

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: *Lost Update***  T1 (User = NHANVIEN): Chỉnh sửa thông tin cá nhân  T2 (User = CHUNHA): Lập hợp đồng thuê/bán nhà có đăng ký tại chi nhánh nơi nhân viên đó làm việc cho khách thuê/mua | | | |
| capnhat\_thongtin\_chunha | **Khóa** | laphopdong\_b2 | **Khóa** |
| ***Input:*** @machunha char(5), @ten NVARCHAR(50), @dchi NVARCHAR(250), @SDT VARCHAR(15), @PASS\_CNHA VARCHAR(50)  ***Output***: Thông tin của chủ nhà được cập nhật | ***Input***: @manha char(5), @manv char(5)  ***Output***: Thông tin nhân viên lập hợp đồng, mã hợp đồng, ngày kết thúc và bắt đầu của hợp đồng; thông tin chủ nhà; thông tin khách hàng |
| SET TRANSACTION ISOLATION  LEVEL **READ COMMITED** | SET TRANSACTION ISOLATION  LEVEL **READ COMMITED** |
| BEGIN TRAN |  |  |  |
| WAITFOR DELAY '0:0:04' |  |  |  |
| B1: Cập nhật tên hoặc địa chỉ hoặc số điện thoại nếu giá trị truyền vào tương ứng khác NULL.  IF @ten is not null  BEGIN  UPDATE CHUNHA SET TEN\_CNHA = @ten WHERE MA\_CNHA = @machunha  END  IF @dchi is not null  BEGIN  UPDATE CHUNHA SET DCHI\_CNHA= @dchi WHERE MA\_CNHA = @machunha  END  IF @SDT IS NOT NULL  BEGIN  UPDATE CHUNHA SET SDT = @SDT WHERE MA\_CNHA = @machunha  END  IF @PASS\_CNHA IS NOT NULL  BEGIN  UPDATE CHUNHA SET PASS\_CNHA = @PASS\_CNHA WHERE MA\_CNHA = @machunha  END | **W(CHUNHA)**  Xin khoá ghi trên bảng CHUNHA với điều kiện MA\_CNHA = @machunha |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: Khai báo và gán giá trị cho ngày bắt đầu và kết thúc của hợp đồng  declare @ngaybd date  declare @ngaykt date  set @ngaybd = getdate()  set @ngaykt = DATEADD(day,2,@ngaybd) | **R(CHUNHA)**  //Không cần xin khoá |
|  |  | B2: Chọn thông tin cá nhân của chủ nhà và khách hàng  select TEN\_CNHA, DCHI\_CNHA, SDT  FROM CHUNHA  WHERE MA\_CNHA = 'CN001'  select TEN\_KH, DCHI\_KH,SDT  FROM KHACHHANG  WHERE MA\_KH = 'KH001' | **R(CHUNHA, KHACHHANG)**  Xin khoá đọc trên bảng CHUNHA, KHACHHANG với điều kiện MA\_CNHA = ‘CN001’, MA\_KH = ‘KH001’ |
| COMMIT |  |  |  |
|  |  | WAITFOR DELAY '0:0:03' |  |
|  |  | B3: Chọn thông tin nhân viên, mã hợp đồng, ngày bắt đầu và kết thúc của hợp đồng  select TEN\_NV, MA\_HD, NGAYBD\_HD, NGAYKT\_HD  FROM NHANVIEN, HOPDONG  WHERE LAMVIEC = (select MA\_CN from NHA where MA\_NHA = @manha)  AND HOPDONG.MA\_NHA = @manha  AND NHANVIEN.MA\_NV = @manv | **R(NHANVIEN, HOPDONG, NHA)**  Xin khoá đọc trên bảng NHANVIEN, HOPDONG, NHA với điều kiện MA\_NHA = @manha, HOPDONG.MA\_NHA = @manha, NHANVIEN.MA\_NV = @manv |
|  |  | COMMIT |  |

Ta có thể hình dung đơn giản đoạn gây lỗi như sau:

A picture containing chart

Description automatically generated

# Sinh viên thực hiện: Ngô Thị Thanh Thảo

## Tình huống 7:

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: *Unrepeatable Data***  T1 (User = KHACHHANG): Tìm nhà dựa trên tiêu chí cụ thể  T2 (User = CHUNHA): Cập nhật, chỉnh sửa thông tin của nhà | | | |
| Tìm nhà dựa trên tiêu chí cụ thể | **Khóa** | Cập nhật, chỉnh sửa thông tin của nhà | **Khóa** |
| ***Input: @GIABAN INTEGER***  ***Output***: Thông tin nhà được tìm kiếm | ***Input***: @MANHA CHAR(5) = NULL,@MA\_LN CHAR(5) = NULL, @GIABAN INTEGER = NULL, @SLP INTEGER = NULL,  @KHUVUC NVARCHAR(30) = NULL, @DUONG NVARCHAR(30) = NULL, @QUAN NVARCHAR(30) = NULL,  @TPHO NVARCHAR(40) = NULL, @NHH DATE = NULL, @TINHTRANG NVARCHAR(100) = NULL, @CHINHANH CHAR(5) = NULL  ***Output***: Thông tin nhà bán được cập nhật |
|  |  |
| BEGIN TRAN |  |  |  |
| B1: Đọc thông tin của nhà thỏa điều kiện  SELECT NHA.MA\_NHA,CHUNHA.TEN\_CNHA AS CHUNHA, NHA.SLPHONG, NHA.VIEW\_NHA, NHABAN.GIABAN, NHA.NGAYDANG, NHA.NGAYHETHAN  FROM NHA, NHABAN, CHUNHA  WHERE NHA.MA\_NHA= NHABAN.MA\_NHA AND NHABAN.GIABAN <= @GIABAN AND NHA.MA\_CNHA = CHUNHA.MA\_CNHA | **R(NHA, NHABAN, CHUNHA)**  Xin khoá đọc trên bảng NHA, NHABAN, CHUNHA với điều kiện NHA.MA\_NHA = NHABAN.MA\_NHA, NHABAN.GIABAN <= @GIABAN AND NHA.MA\_CNHA = CHUNHA.MA\_CNHA |  |  |
| WAITFOR DELAY '00:00:06' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: Cập nhật thông tin của nhà bán  IF @MA\_LN IS NOT NULL  BEGIN  UPDATE NHA SET MALN = @MA\_LN WHERE MA\_NHA = @MANHA  END  IF @TINHTRANG IS NOT NULL  BEGIN  UPDATE NHA SET TINHTRANG = @TINHTRANG WHERE MA\_NHA = @MANHA  END  IF @GIABAN IS NOT NULL  BEGIN  UPDATE NHABAN SET GIABAN = @GIABAN WHERE MA\_NHA = @MANHA  END  IF @KHUVUC IS NOT NULL  BEGIN  UPDATE DCHINHA SET KHUVUC = @KHUVUC WHERE MA\_NHA = @MANHA  END  IF @DUONG IS NOT NULL  BEGIN  UPDATE DCHINHA SET DUONG = @DUONG WHERE MA\_NHA = @MANHA  END  IF @QUAN IS NOT NULL  BEGIN  UPDATE DCHINHA SET QUAN = @QUAN WHERE MA\_NHA = @MANHA  END  IF @TPHO IS NOT NULL  BEGIN  UPDATE DCHINHA SET TPHO = @TPHO WHERE MA\_NHA = @MANHA  END  IF @NHH IS NOT NULL  BEGIN  UPDATE NHA SET NGAYHETHAN = @NHH WHERE MA\_NHA = @MANHA  END  IF @SLP IS NOT NULL  BEGIN  UPDATE NHA SET SLPHONG = @SLP WHERE MA\_NHA = @MANHA  END  IF @CHINHANH IS NOT NULL  BEGIN  UPDATE NHA SET MA\_CN = @CHINHANH WHERE MA\_NHA = @MANHA  END | **W(NHA)**  Xin khóa ghi trên bảng NHA với điều kiện NHA.MA\_NHA = @MANHA |
|  |  | COMMIT |  |
| B2: In số lượng nhà thỏa điều kiện  PRINT N'Số lượng nhà thỏa điều kiện trên là'  SELECT COUNT(\*) AS SOLUONGNHATHOADK  FROM NHA,NHABAN,CHUNHA  WHERE NHA.MA\_NHA=NHABAN.MA\_NHA AND NHABAN.GIABAN <=@GIABAN AND NHA.MA\_CNHA=CHUNHA.MA\_CNHA | **R(NHA, NHABAN, CHUNHA)**  Xin khoá đọc trên bảng NHA, NHABAN, CHUNHA với điều kiện NHA.MA\_NHA = NHABAN.MA\_NHA, NHABAN.GIABAN <= @GIABAN AND NHA.MA\_CNHA = CHUNHA.MA\_CNHA |  |  |
| COMMIT |  |  |  |

Ta có thể hình dung đơn giản đoạn gây lỗi như sau:

