**d**

BÁO CÁO ĐỒ ÁN THỰC HÀNH

MÔN HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU

GVHD: HỒ THỊ HOÀNG VY

# **THÔNG TIN NHÓM**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **MSSV** | **Họ tên** | **Công việc** | **% Hoàn thành** |
| **1** | **18127257** | **Trần Huy Vũ** | **2 lỗi tranh chấp** | **100%** |
| **2** | **18127215** | **Quách Phú Thành** | **2 lỗi tranh chấp** | **100%** |
| **3** | **18127234** | **Trần Ngọc Bảo Trân** | **2 lỗi tranh chấp** | **100%** |
| **4** | **18127188** | **Trần Thị Thế Phương** | **2 lỗi tranh chấp** | **100%** |
| **5** | **18127094** | **Trịnh Quang Hà** | **2 lỗi tranh chấp** | **100%** |

# XỬ LÝ TÌNH HUỐNG TRANH CHẤP

# Sinh viên thực hiện: Trần Huy Vũ - 18127257

## Tình huống 1: Admin đang thay đổi thông tin của 1 căn nhà, nhân viên xem căn nhà đó

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: Dirty read**  T1 (User = admin): thực hiện việc thay đổi thông tin căn nhà có mã nhà là @nha\_id  T2 (User = nhân viên): thực hiện xem thông tin nhà với mã nhà là @nha\_id giống với admin | | | |
| **ttnha\_\_insert** | **Khóa** | **NV\_XemThongTinNha** | **Khóa** |
| ***Input:*** Những thông tin của căn nhà  ***Output***: Thông tin căn nhà sau khi đã thêm | ***Input:*** Mã nhà  ***Output***: Thông tin căn nhà |
| SET TRANSACTION ISOLATION  LEVEL READ COMMITED | SET TRANSACTION ISOLATION  LEVEL READ COMMITED |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra mã nhà có tồn tại hay không  if exists(select \* from NHA where nha\_id=@nha\_id)  begin  print N'Nha' + cast(@nha\_id as varchar(6)) + N' Đã Tồn Tại'  rollback tran  return 1  end | **R1(NHA)**  Xin khoá đọc trên bảng NHA với đk nha\_id = @nha\_id |  |  |
| B2: Kiểm tra mã loại nhà có hợp lệ hay không  if @nha\_idloai is null or @nha\_idloai = ''  begin  print N'Loai Nha Không Được Trống'  rollback tran  return 1  end |  |  |  |
| B3: Kiểm tra mã chủ nhà có hợp lệ hay không  if @nha\_idchu is null or @nha\_idchu = ''  begin  print N'Chủ nhà không được bỏ trống'  rollback tran  return 1  end |  |  |  |
| B4: Kiểm tra mã chi nhánh có hợp lệ hay không  if @nha\_idcn is null or @nha\_idcn = ''  begin  print N'ID Chi nhánh không được bỏ trống'  rollback tran  return 1  end |  |  |  |
| B5: Kiểm tra mã nhân viên có hợp lệ hay không  if @nha\_idnv is null or @nha\_idnv = ''  begin  print N'ID Nhân viên không được bỏ trống'  rollback tran  return 1  end |  |  |  |
| if @nha\_slphong is null or @nha\_slphong = ''  begin  print N'So Luong Phong Không Được Trống'  rollback tran  return 1  end |  |  |  |
| if @nha\_ttrang is null or @nha\_ttrang = ''  begin  print N'Tinh Trang Nha Không Được Trống'  rollback tran  return 1  end |  |  |  |
| if @nha\_duong is null or @nha\_duong = ''  begin  print N'Dia Chi Không Được Trống'  rollback tran  return 1  end |  |  |  |
| Chỉnh sửa thông tin của ngôi nhà  update NHA  set  nha\_slphong=@nha\_slphong,  nha\_ttrang=@nha\_ttrang,  nha\_duong=@nha\_duong,  nha\_quan=@nha\_quan,  nha\_kvuc=@nha\_kvuc,  nha\_tp=@nha\_tp,  nha\_giat=@nha\_giat,  nha\_giab=@nha\_giab,  nha\_dkban=@nha\_dkban,  nha\_idloai=@nha\_idloai,  nha\_idchu=@nha\_idchu,  nha\_idcn=@nha\_idcn,  nha\_idnv=@nha\_idnv,  nha\_ndang=@nha\_ndang,  nha\_hethan=@nha\_hethan  where nha\_id=@nha\_id | **X(NHA)**  Xin khoá ghi trên bảng NHA để thay đổi thông tin nhà  Giữ khóa cho đến hết giao tác |  |  |
| WAITFOR DELAY '00:00:05' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: Kiểm tra mã nhà có tồn tại hay không  if not exists(select \* from NHA where nha\_id = @nha\_id)  begin  print N'Nha' + cast(@nha\_id as varchar(6)) + N' Khong Tồn Tại'  rollback tran  return 1  end | **R2(NHA)**  Xin khoá đọc trên bảng NHA với đk nha\_id = @nha\_id (tuy nhiên lúc này, tran1 đã giữ khóa ghi trên bảng NHA cho nên tran2 phải chờ đến khi tran1 thực hiện xong) |
|  |  | B2: Hiển thị thông tin nhà trùng với mã vừa rồi  SELECT nha\_id 'Mã nhà',nha\_slphong 'Số lượng phòng', nha\_ttrang 'Tình trạng nhà', nha\_duong 'Đường', nha\_quan 'Quận', nha\_kvuc 'Khu vực', nha\_tp 'Thành phố'  FROM NHA  WHERE nha\_id = @nha\_id |  |
|  |  | COMMIT |  |
| ROLLBACK TRAN nhà mới có ngày đăng trước ngày hết hạn, vi phạm ràng buộc ngày hết hạn phải sau ngày đăng nên rollback |  |  |  |
| BEGIN CATCH |  |  |  |
| print N'LỖI HỆ THỐNG'  ROLLBACK TRAN  return 1 |  |  |  |
| END CATCH |  |  |  |
| COMMIT |  |  |  |

**Giải thích xử lý tranh chấp**: Đầu tiên admin tiến hành sửa thông tin của ngôi nhà trong bảng NHA. Lúc đó, nhân viên cũng tiến hành xem thông tin ngôi nhà đó, tuy nhiên vì một số sơ sót nên admin đã nhập dữ liệu ngày đăng nhà sau ngày hết hạn (vô lý, điều này vi phạm ràng buộc ngày hết hạn phải sau ngày đăng nhà của hệ thống) dẫn đến rollback toàn bộ dữ liệu của ngôi nhà đó, trở về trạng thái ban đâu. Vì đã set mức cô lập READ COMMITED cho giao tác thứ hai (giao tác nhân viên xem nhà) nên nó phải đợi giao tác thứ nhất (admin chỉnh sửa nhà) hoàn thành mới được thực hiện đọc trên bảng NHA 🡪 xử lý được lỗi Dirty Read

## Tình huống 2: Admin đang tìm kiếm nhân viên theo giới tính nam sau đó chọn ra 1 nhân viên để nhờ giúp đỡ 1 việc gì đó, nhưng ở một máy tính khác, một admin khác xóa thông tin nhân viên đó đi

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR02: UNREPTABLE READ**  T1 (User = admin): thực hiện việc tìm kiếm thông tin một nhân viên theo giới tính  T2 (User = admin): thực hiện xem xóa thông tin nhân viên với @MaNV | | | |
| **AD\_XemThongTinNhanVien** | **Khóa** | **AD\_XoaNhanVien** | **Khóa** |
| ***Input:*** Mã nhân viên cần tìm  ***Output***: Thông tin nhân viên | ***Input:*** Mã nhân viên cần xóa  ***Output***: bảng nhân viên không còn nhân viên đó nữa |
| SET TRANSACTION ISOLATION  LEVEL REPEATABLE READ | SET TRANSACTION ISOLATION  LEVEL REPEATABLE READ |
| BEGIN TRAN |  |  |  |
| B1: Đọc thông tin nhân viên theo giới tính nam  begin  select \*  from NHAN\_VIEN  where nv\_gtinh = ‘Nam’  end; | **R1(A)**  Xin khoá đọc đến hết giao tác |  |  |
| WAITFOR DELAY '00:00:20' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | BEGIN TRY |  |
|  |  | B1: Kiểm tra mã nhân viên có tồn tại hay không  IF NOT EXISTS (SELECT \* FROM NHAN\_VIEN WHERE nv\_id=@MaNV)  BEGIN  PRINT N'NV' + CAST(@MaNV AS VARCHAR(3)) + N' Không Tồn Tại'  ROLLBACK TRAN  RETURN 1  END | **R2(A)**  Xin khoá đọc trên bảng NHAN\_VIEN với đk nv\_id = @MaNV |
|  |  | B2: Kiểm tra nhân viên này đã đăng nhà hay chưa  IF EXISTS (SELECT \* FROM NHA WHERE nha\_idnv = @MaNV)  BEGIN  PRINT N'NV' + CAST(@MaNV AS VARCHAR(3)) + N' Đã Thực Hiện Đăng Nhà, Không Xóa Được'  ROLLBACK TRAN  RETURN 1  END  DELETE FROM NHAN\_VIEN WHERE nv\_id = @MaNV |  |
|  |  | END TRY |  |
|  |  | B3: Xóa nhân viên  DELETE FROM NHAN\_VIEN WHERE nv\_id = @MaNV | **W(A)**  Xin khoá ghi trên bảng NHAN\_VIEN với đk nv\_id = @MaNV |
|  |  | WAITFOR DELAY '00:00:20' |  |
|  |  | COMMIT |  |
| B2: Đọc thông tin nhân viên X nào đó  exec TimKiemNhanVien 1 | **R1(A)**  Xin khoá đọc trên bảng NHAN\_VIEN đến hết giao tác |  |  |

## **Giải thích xử lý tranh chấp:** Đầu tiên, admin (tạm gọi là admin số 1) tìm ra tất cả nhân viên có giới tính là nam, từ đó xem được mã nhân viên mà mình muốn giao đi công tác. Trong quá trình xem và chọn lựa đó. Thì một admin khác (admin số 2) tại cơ sở khác tiến hành xóa đi thông tin nhân viên mà admin số 1 dự tính chọn lựa. Vì đã set mức cô lập là REPEATABLE READ cho việc đọc dữ liệu (giao tác thứ nhất – admin tìm kiếm nhân viên) nên giao tác thứ hai (admin xóa nhân viên) phải đợi giao tác thứ nhất hoàn thành mới thực hiện được 🡪 xử lý được lỗi Unrepeatable read

# Sinh viên thực hiện: Trần Thị Thế Phương - 18127188

## Xem thông tin của khách hàng (tìm kiếm khách hàng theo tên). Nhân viên thêm một khách hàng mới cùng tên với khách hàng cũ.

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR02: Phamtom read**  T1 (User = nhân viên 1): xem thông tin của khách hàng tên A  T2 (User = nhân viên 2): thêm một khách hàng mới cũng có tên A | | | |
| **sp xemThongTinKhachHang** | **Khóa** | **sp\_ThemMotKhachHang** | **Khóa** |
| ***Input:*** Tên khách hàng  ***Output***: Danh sách thông tin khách hàng | ***Input***: ID, tên khách hàng  ***Output***: Danh sách khách hàng |
| SET TRANSACTION ISOLATION  LEVEL **SERIALIZABLE** |  |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra tên KH có tồn tại  exec ttkh\_view N'Nguyen Van D'  --  Hàm ttkh\_view  IF NOT EXISTS (SELECT \* FROM KHACH\_HANG WHERE kh\_ten = @tenKH)  BEGIN  PRINT 'KH' + CAST (@tenKH AS VARCHAR(3)) + N' Không Tồn Tại'  ROLLBACK TRAN  RETURN  END | **R(A)**  **//Xin khóa S và giữ cho đến khi hết giao tác** |  |  |
| WAITFOR DELAY '00:00:20' |  |  |  |
|  |  | B1: Kiểm tra thông tin khách hàng đã tồn tại  DECLARE @RETRY\_COUNT\_CURRENT INT  DECLARE @RETRY\_COUNT\_MAXIMUM INT  DECLARE @ERROR\_NUM INT  DECLARE @ERROR\_MSG NVARCHAR(MAX)    SET @ERROR\_NUM = 0  SET @RETRY\_COUNT\_CURRENT = 0  SET @RETRY\_COUNT\_MAXIMUM = 3    WHILE @RETRY\_COUNT\_CURRENT < @RETRY\_COUNT\_MAXIMUM  BEGIN  BEGIN TRANSACTION  BEGIN TRY  exec ttkh\_\_insert 31,N'Nguyen Van D',N'HCM','0881514415','CN0002'  COMMIT    BREAK  END TRY  BEGIN CATCH  SELECT @ERROR\_NUM = ERROR\_NUMBER(), @ERROR\_MSG = ERROR\_MESSAGE()  PRINT @ERROR\_MSG  PRINT @ERROR\_NUM  ROLLBACK  SET @RETRY\_COUNT\_CURRENT = @RETRY\_COUNT\_CURRENT + 1  WAITFOR DELAY '00:00:01'  CONTINUE  END CATCH;  END  ---Hàm ttkh\_insert  if exists (select \* from KHACH\_HANG where kh\_id=@kh\_id)  begin  print N'Khach hang' + cast (@kh\_id as varchar (3)) + N' Đã Tồn Tại'  rollback tran  return 1  end  if @kh\_ten is null or @kh\_ten = ''  begin  print N'Tên Khach Hang Không Được Trống'  rollback tran  return 1  end  IF LEN(@kh\_sdt) <> 10  begin  print N'SDT Không Hợp Lệ'  rollback tran  return 1  end  if @kh\_dchi is null or @kh\_ten = ''  begin  print N'Dia Khach Hang Không Được Trống'  rollback tran  return 1  end  if not exists (select \* from dbo.CHI\_NHANH where cn\_id=@kh\_idcn)  begin  print N'Chi Nhánh '+ @kh\_idcn + N' Không Tồn Tại'  rollback tran  return 1  end | **R(A)**  **//Xin khoá S trên bảng KHACH\_HANG với điều kiện kh\_id = @kh\_id**  **Giữ khóa cho đến hết giao tác** |
|  |  | B2: Thêm thông tin khách hàng vào bảng  insert into dbo.KHACH\_HANG (kh\_id,kh\_ten,kh\_dchi,kh\_sdt,kh\_idcn)  values (@kh\_id,@kh\_ten,@kh\_dchi,@kh\_sdt,@kh\_idcn) | **W(KHACH\_HÀNG)**  **//Xin khoá X trên bảng KHACH\_HANG** |
|  |  |  |  |
|  |  | COMMIT |  |
| B2: Hiển thị thông tin khách hàng  select \*  from KHACH\_HANG kh  where kh\_ten =N’A’ | **R(KHACH\_HANG)**  **//Xin khóa S và giữ cho đến hết giao tác** |  |  |
| waitfor delay '00:00:20'  commit transaction |  |  |  |

## **Giải thích xử lý tranh chấp:** T1 đang thực hiện khóa S trên khách hàng có tên A (đang ở mức cô lập SERIALIAZABLE). Do đó, dù T2 có ghi trên dữ liệu KH A thì T1 vẫn giữ khóa và chạy cho đến kết thúc giao tác. Kết quả trả về là dữ liệu tìm kiếm ban đầu (khi chưa insert).

## Xóa thông tin của nhà

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR02: Conversion deadlock**  T1 (User = Admin): chỉnh sửa thông tin của nhà A  T2 (User = Nhân viên): chỉnh sửa thông tin của nhà A | | | |
| **sp** | **Khóa** | **sp\_ThemMotKhachHang** | **Khóa** |
| ***Input:*** ID nhân viên  ***Output***: Thông tin nhân viên sau khi chỉnh sửa | ***Input***: ID, tên khách hàng  ***Output***: Thông tin nhân viên sau khi chỉnh sửa |
| SET DEADLOCK\_PRIORITY NORMAL  BEGIN TRAN  SET TRANSACTION ISOLATION LEVEL SERIALIZABLE  exec ttnha\_view 'BT0006'    --Kiểm tra thông tin nhà  if exists(select \* from NHA where nha\_id=@nha\_id)  begin  print N'Nha' + cast(@nha\_id as varchar(6)) + N' Đã Tồn Tại'  rollback tran  return 1  end | **R(A)**  **//Xin khóa S và giữ cho đến khi hết giao tác** |  |  |
| WAITFOR DELAY '00:00:20' |  |  |  |
|  |  | SET DEADLOCK\_PRIORITY LOW  DECLARE @RETRY\_COUNT\_CURRENT INT  DECLARE @RETRY\_COUNT\_MAXIMUM INT  DECLARE @ERROR\_NUM INT  DECLARE @ERROR\_MSG NVARCHAR(MAX)    SET @ERROR\_NUM = 0  SET @RETRY\_COUNT\_CURRENT = 0  SET @RETRY\_COUNT\_MAXIMUM = 3    WHILE @RETRY\_COUNT\_CURRENT < @RETRY\_COUNT\_MAXIMUM  BEGIN  BEGIN TRANSACTION  BEGIN TRY  exec ttnha\_view 'BT0006'  exec ttnha\_\_update 'BT0006',6,N'Còn trống',N'Đường số 1',N'Quận 8\*\*\*',N'Trung tâm',N'TP HCM',36000000.0,NULL,NULL,'BT',5,'CN0002',8,'2020-10-21','2020-11-21'    COMMIT    BREAK  END TRY  BEGIN CATCH  SELECT @ERROR\_NUM = ERROR\_NUMBER(), @ERROR\_MSG = ERROR\_MESSAGE()  PRINT @ERROR\_MSG  PRINT @ERROR\_NUM  ROLLBACK  SET @RETRY\_COUNT\_CURRENT = @RETRY\_COUNT\_CURRENT + 1  WAITFOR DELAY '00:00:05'  CONTINUE  END CATCH;  END | **R(A)**  **//Xin khoá S trên bảng NHA với điều kiện @nha\_id=nha\_id Giữ khóa cho đến hết giao tác** |
| B2: Thêm thông tin nhà  exec ttnha\_\_update 'BT0006',6,N'Còn trống',N'Đường số 1',N'Quận 8\*\*',N'Trung tâm',N'TP HCM',36000000.0,NULL,NULL,'BT',5,'CN0002',8,'2020-10-21','2020-11-21'  commit transaction |  |  |  |

## **Giải thích xử lý tranh chấp tranh chấp:** Vì T1 đang giữ khóa S trên nhà A nên nó chờ T2 commit để lấy khóa X để ghi trên row nhà A, nhưng T2 cũng đang giữ khóa S trên nhà A, nó cũng chờ T1 commit để được ghi tên row đó. Kết quả cả hai giao tác cùng chờ nhau 🡪 conversion deadlock => Sử dụng mức cô lập SERIALIZABLE để T1 thực hiện hết giao tác, trả khóa thì T2 sẽ thực hiện 🡪 Giải quyết định deadlock

# Sinh viên thực hiện: Trần Ngọc Bảo Trân - 18127234

## Tình huống 1: Khách hàng muốn tìm căn nhà có giá cả phù hợp, yêu cầu này được gửi đến nhân viên. Nhân viên đó bắt đầu tìm kiếm một căn nhà theo nhu cầu về giá nhà của khách hàng. Trong lúc đó, một nhân viên khác tiến hành thêm nhà bán.

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR02: Phamtom read**  T1 (User = nhân viên): thực hiện tìm kiếm căn nhà theo nhu cầu về giá nhà.  T2 (User = nhân viên): thực hiện việc thêm thông tin về nhà bán. | | | |
| **sp TimKiemYCLoai** | **Khóa** | **sp\_ThongKeChuyenXe-CoDieuKien** | **Khóa** |
| ***Input:*** Giá nhà  ***Output***: Danh sách nhà có loại tương ứng | ***Input***: Thông tin căn nhà  ***Output***: Căn nhà được thêm vào bảng |
| SET TRANSACTION ISOLATION  LEVEL **SERIALIZABLE** | SET TRANSACTION ISOLATION  LEVEL **REPEATABLE READ** |
| BEGIN TRAN |  |  |  |
| B1: Đọc loại nhà cần tìm  begin try  select nh\_nha\_id  from NHA nh  where @Gia < nh.nha\_giab  end try | **R(A)**  **//Xin khóa S và giữ cho đến khi hết giao tác** |  |  |
| WAITFOR DELAY '00:00:15' |  |  |  |
|  |  | B1: Kiểm tra thông tin của căn nhà  if not exists(select \* from NHA where nha\_id=@nha\_id)  begin  print N'Nha' + cast(@nha\_id as varchar(6)) + N' Khong Tồn Tại'  rollback tran  return 1  end end | **R(A)**  **//Xin khoá S trên bảng NHA với điều kiện nha\_id = @nha\_id**  **Giữ khóa cho đến hết giao tác** |
|  |  | B2: Kiểm tra thông tin của căn nhà không được trống  if @nha\_idloai is null or @nha\_idloai = ''  begin  print N'Loai Nha Không Được Trống'  rollback tran  return 1  end  if @nha\_idchu is null or @nha\_idchu = ''  begin  print N'Chủ nhà không được bỏ trống'  rollback tran  return 1  end  if @nha\_idcn is null or @nha\_idcn = ''  begin  print N'ID Chi nhánh không được bỏ trống'  rollback tran  return 1  end  if @nha\_idnv is null or @nha\_idnv = ''  begin  print N'ID Nhân viên không được bỏ trống'  rollback tran  return 1  end  if @nha\_slphong is null or @nha\_slphong = ''  begin  print N'So Luong Phong Không Được Trống'  rollback tran  return 1  end  if @nha\_ttrang is null or @nha\_ttrang = ''  begin  print N'Tinh Trang Nha Không Được Trống'  rollback tran  return 1  end  if @nha\_duong is null or @nha\_duong = ''  begin  print N'Dia Chi Không Được Trống'  rollback tran  return 1  end  if @nha\_quan is null or @nha\_quan = ''  begin  print N'Dia Chi Không Được Trống'  rollback tran  return 1  end  if @nha\_kvuc is null or @nha\_kvuc = ''  begin  print N'Dia Chi Không Được Trống'  rollback tran  return 1  end | **R(ChuyenDi)**  **//Xin khoá S trên bảng NHA cho đến khi hết giao tác** |
|  |  | B2: Thêm thông tin nhà vào bảng  insert into NHA (nha\_id,nha\_slphong,  nha\_ttrang,nha\_duong, nha\_quan,nha\_kvuc,  nha\_tp,nha\_giat,nha\_giab,nha\_dkban,nha\_idloai,  nha\_idchu,nha\_idcn,nha\_idnv,  nha\_ndang,nha\_hethan)  values (@nha\_id,@nha\_slphong,@nha\_ttrang,  @nha\_duong, @nha\_quan,@nha\_kvuc,@nha\_tp,  @nha\_giat,@nha\_giab,@nha\_dkban,@nha\_idloai,  @nha\_idchu,@nha\_idcn,@nha\_idnv,  @nha\_ndang,@nha\_hethan) | **W(NHA)**  **//Xin khoá X trên bảng NHA** |
|  |  | WAITFOR DELAY '00:00:20' |  |
|  |  | COMMIT |  |
| B2: Đọc toàn bộ thông tin nhà  select \*  from NHA nh  where @Gia < nh.nha\_giab | **R(NHA)**  **//Xin khóa S và giữ cho đến hết giao tác** |  |  |

## **Giải thích xử lý tranh chấp:** Bước đầu tiên nhân viên sẽ nhập giá mà khách hàng muốn mua, từ đó xác định mã các căn nhà có giá cả phù hợp. Sau đó, nhân viên đó tiến hành các bước nghiệp vụ khác. Trong quá trình đó, một nhân viên khác lại tiến hành thêm thông tin căn nhà có giá cả phù hợp với khoảng mà khách hàng cần. Nhưng là bởi vì đã cài đặt mức cô lặp là Serializable cho việc đọc dữ liệu, nên tiến trình thêm thông tin nhà sẽ phải chờ cho đến khi việc đọc thông tin nhà hoàn thành thì khi đó việc ghi mới được thực hiện 🡺 Hai lần đọc dữ liệu điều không bị thay đổi 🡪 Lỗi phantom được sửa

## Tình huống 2: Một nhân viên đang tìm kiếm nhà theo nhu cầu loại nhà của khách hàng. Trong lúc đó, một nhân viên khác tiến hành xóa nhà bán.

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR03: Unrepeatable read**  T1 (User = nhân viên): thực hiện tìm kiếm căn nhà theo nhu cầu về khu vực.  T2 (User = nhân viên): thực hiện việc xóa thông tin về nhà bán. | | | |
| **sp TimKiemYCLoai** | **Khóa** | **sp\_ThongKeChuyenXe-CoDieuKien** | **Khóa** |
| ***Input:*** Loại nhà  ***Output***: Danh sách nhà có loại tương ứng | ***Input***: Thông tin căn nhà  ***Output***: Căn nhà đã xóa |
| SET TRANSACTION ISOLATION  LEVEL **REPEATABLE READ** | SET TRANSACTION ISOLATION  LEVEL… |
| BEGIN TRAN |  |  |  |
| B1: Đọc loại nhà  begin try  select \*  from NHA nh  where nh.nha\_kvuc like @KVuc  end try | **R(A)**  **//Xin khóa S và giữ cho đến khi hết giao tác** |  |  |
| WAITFOR DELAY '00:00:15' |  |  |  |
|  |  | B1: Kiểm tra thông tin của căn nhà  if not exists (select \*from NHA where nha\_id=@nha\_id)  begin  print N'Nha' + cast(@nha\_id as varchar(6)) + N'Khong ton tai'  rollback tran  return 1  end | **R(A)**  **//Xin khoá S trên bảng NHA với điều kiện nha\_id = @nha\_id**  **Giữ khóa cho đến hết giao tác** |
|  |  | B2: Xóa thông tin căn nhà  delete from NHA where nha\_id=@nha\_id | **W(NHA)**  **//Xin khoá X trên bảng NHA** |
|  |  | WAITFOR DELAY '00:00:20' |  |
|  |  | COMMIT |  |
| B2: Đọc danh sách nhà  select \*  from NHA nh  where nh.nha\_quan like @KVuc | **R(NHA)**  **//Xin khóa S cho đến hết giao tác** |  |  |

**Giải thích xử lý tranh chấp:** Bước đầu tiên nhân viên sẽ nhập khu vực mà khách hàng muốn tìm mua nhà, từ đó xác định mã các căn nhà có khu vực. Sau đó, nhân viên đó tiến hành các bước nghiệp vụ khác. Trong quá trình đó, một nhân viên khác lại tiến hành xóa thông tin căn nhà có khu vực phù hợp với khoảng mà khách hàng cần. Nhưng là bởi vì đã cài đặt mức cô lặp là Repeatable read cho việc đọc dữ liệu, nên tiến trình xóa thông tin nhà sẽ phải chờ cho đến khi việc đọc thông tin nhà hoàn thành thì khi đó việc ghi mới được thực hiện 🡺 Hai lần đọc dữ liệu điều không bị thay đổi 🡪 Lỗi Unrepeatable được sửa

# Sinh viên thực hiện: Trịnh Quang Hà - 18127094

## Tình huống 1: Admin tiến hành cập nhật thông tin của 1 căn nhà. Cùng lúc đó, nhân viên tiến hành lập hợp đồng cho thuê căn nhà đó cho khách hàng.

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: *Lost update***  T1 (User = Admin): thực hiện cập nhật thông tin cho căn nhà có mã nhà @nha\_id  T2 (User = Nhân viên): thực hiện lập hợp đồng thuê nhà căn nhà có mã nhà @MaNha cho khách hàng (@MaNha = @nha\_id) | | | |
| **ttnha\_\_update** | **Khóa** | **NV\_LapHopDong** | **Khóa** |
| ***Input:*** Các thuộc tính của căn nhà  ***Output***: Cập nhật lại các thuộc tính của căn nhà. | ***Input***: Thông tin khách hàng, chủ nhà, căn nhà, ngày thuê, ngày hết hạn  ***Output***: Lập ra 1 hợp đồng thuê nhà |
| SET TRANSACTION ISOLATION  LEVEL REAPEATABLE READ | SET TRANSACTION ISOLATION  LEVEL READ COMMITED |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra mã nhà có tồn tại hay không  if not exists(select \* from NHA where nha\_id = @nha\_id)  begin  print N'Nha' + cast(@nha\_id as varchar(6)) + N' Khong Tồn Tại'  rollback tran  return 1  end | **R(NHA)**  Xin khoá đọc trên bảng NHA với đk nha\_id = @nha\_id  Và giữ đến khi thực hiện hết giao tác |  |  |
| B2: Kiểm tra mã loại nhà có hợp lệ hay không  if @nha\_idloai is null or @nha\_idloai = ''  begin  print N'Loai Nha Không Được Trống'  rollback tran  return 1  end |  |  |  |
| B3: Kiểm tra mã chủ nhà có hợp lệ hay không  if @nha\_idchu is null or @nha\_idchu = ''  begin  print N'Chủ nhà không được bỏ trống'  rollback tran  return 1  end |  |  |  |
| B4: Kiểm tra mã chi nhánh có hợp lệ hay không  if @nha\_idcn is null or @nha\_idcn = ''  begin  print N'ID Chi nhánh không được bỏ trống'  rollback tran  return 1  end |  |  |  |
| B5: Kiểm tra mã nhân viên có hợp lệ hay không  if @nha\_idnv is null or @nha\_idnv = ''  begin  print N'ID Nhân viên không được bỏ trống'  rollback tran  return 1  end |  |  |  |
| if @nha\_slphong is null or @nha\_slphong = ''  begin  print N'So Luong Phong Không Được Trống'  rollback tran  return 1  end |  |  |  |
| if @nha\_ttrang is null or @nha\_ttrang = ''  begin  print N'Tinh Trang Nha Không Được Trống'  rollback tran  return 1  end |  |  |  |
| if @nha\_duong is null or @nha\_duong = ''  begin  print N'Dia Chi Không Được Trống'  rollback tran  return 1  end |  |  |  |
| if @nha\_quan is null or @nha\_quan = ''  begin  print N'Dia Chi Không Được Trống'  rollback tran  return 1  end |  |  |  |
| if @nha\_kvuc is null or @nha\_kvuc = ''  begin  print N'Dia Chi Không Được Trống'  rollback tran  return 1  end |  |  |  |
| WAITFOR DELAY '00:00:05' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: Khai báo các biến cần thiết và xác định loại hợp đồng từ ngày thông tin ngày hết hạn  DECLARE @MaHD CHAR(6) = dbo.AUTO\_IDHD()  DECLARE @LoaiHD NVARCHAR(6) = N'Thuê'  DECLARE @ThoiGianThue INT  DECLARE @TongTien FLOAT  IF (@NgayHetHan IS NULL)  SET @LoaiHD = N'Bán' |  |
|  |  | B2: Xác định tổng tiền của hợp đồng  IF (@LoaiHD = N'Bán')  BEGIN  SET @TongTien = (SELECT nha\_giab FROM NHA WHERE nha\_id=@MaNha)  END  ELSE  BEGIN  SET @ThoiGianThue = DATEDIFF(MONTH,@NgayThueBan,@NgayHetHan)  SET @TongTien = (SELECT nha\_giat \* @ThoiGianThue FROM NHA WHERE nha\_id=@MaNha)  END | **R(NHA)**  Xin khoá đọc trên bảng NHA với đk nha\_id = @MaNha |
|  |  | B3: Kiểm tra khách hàng có hợp lệ không  IF NOT EXISTS(SELECT \* FROM KHACH\_HANG WHERE kh\_id=@MaKH)  BEGIN  PRINT N'Khách Hàng ' + CAST(@MaKH AS VARCHAR(3)) + ' Chưa Có Trong Danh Sách Khách Hàng'  ROLLBACK TRAN  RETURN 1  END | **R(KHACH\_HANG)**  Xin khoá đọc trên bảng KHACH\_HANG với đk kh\_id=@MaKH |
|  |  | B4: Kiểm tra khách hàng có là chủ nhà hay không  BEGIN  PRINT N'Khách Hàng ' + CAST(@MaKH AS VARCHAR(3)) + ' Không Hợp Lệ Vì Đây Là Chủ Nhà'  ROLLBACK TRAN  RETURN 1  END |  |
|  |  | B4: Kiểm tra khách hàng có xem nhà cần thuê chưa  IF @MaKH NOT IN (SELECT x\_idkh FROM XEM\_NHA WHERE x\_idnha = @MaNha)  BEGIN  PRINT N'Khách Hàng ' + CAST(@MaKH AS VARCHAR(3)) + N' Chưa Xem Nhà'  ROLLBACK TRAN  RETURN 1  END | **R(XEM\_NHA)**  Xin khoá đọc trên bảng XEM\_NHA với đk  x\_idnha = @MaNha |
|  |  | B5: Kiểm tra chủ nhà và nhà có hợp lệ hay không  IF NOT EXISTS(SELECT \* FROM NHA WHERE nha\_id=@MaNha AND nha\_idchu=@MaChuNha)  BEGIN  PRINT N'Thông Tin Chủ Nhà Và Nhà Không Hợp Lệ'  ROLLBACK TRAN  RETURN 1  END | **R(NHA)**  Xin khoá đọc trên bảng NHA với đk nha\_id=@MaNha AND nha\_idchu=@MaChuNha |
|  |  | B5: Kiểm tra tình trạng nhà có đang Còn trống hay không  IF (SELECT nha\_ttrang FROM NHA WHERE nha\_id = @MaNha) IN (N'Đã cho thuê',N'Đã bán')  BEGIN  PRINT N'Nhà đã được cho thuê/bán'  ROLLBACK TRAN  RETURN 1  END | **R(NHA)**  Xin khoá đọc trên bảng NHA với đk nha\_id = @MaNha |
|  |  | B5: Kiểm tra ngày lập hợp đồng có hợp lệ hay không  IF (@NgayThueBan < (SELECT nha\_ndang FROM NHA WHERE nha\_id=@MaNha) OR @NgayThueBan > (SELECT nha\_hethan FROM NHA WHERE nha\_id=@MaNha) OR @NgayThueBan < (SELECT x\_ngxem FROM XEM\_NHA WHERE x\_idkh = @MaKH AND x\_idnha = @MaNha))  BEGIN  PRINT N'Ngày Lập Hợp Đồng Không Hợp Lệ'  ROLLBACK TRAN  RETURN 1  END | **R(NHA)**  Xin khoá đọc trên bảng NHA với đk nha\_id = @MaNha  **R(XEM\_NHA)**  Xin khoá đọc trên bảng XEM\_NHA với đk  x\_idkh = @MaKH AND x\_idnha = @MaNha |
|  |  | B6: Kiểm tra ngày hết hạn lập hợp đồng có hợp lệ hay không (cùng thứ tự ngày trong tháng)  IF (@NgayHetHan <= @NgayThueBan OR DAY(@NgayHetHan) <> DAY(@NgayThueBan))  BEGIN  PRINT N'Ngày Hết Hạn Hợp Đồng Không Hợp Lệ'  ROLLBACK TRAN  RETURN 1  END |  |
|  |  | B7: Thêm thông tin lập hợp đồng vào bảng HOP\_DONG  INSERT INTO HOP\_DONG(hd\_id,hd\_idchu,hd\_idnha,hd\_idkh,hd\_ngay,hd\_ngayhethan,hd\_loai,hd\_tongtien)  VALUES(@MaHD,@MaChuNha,@MaNha,@MaKH,@NgayThueBan,@NgayHetHan,@LoaiHD,@TongTien) | **X(HOP\_DONG)**  Xin khoá ghi trên bảng HOP\_DONG để thêm 1 hợp đồng mới |
|  |  | B8: Cập nhật tình trạng nhà cho căn nhà vừa lập hợp đồng  IF @LoaiHD = N'Thuê'  UPDATE NHA SET nha\_ttrang = N'Đã cho thuê' WHERE nha\_id = @MaNha  IF @LoaiHD = N'Bán'  UPDATE NHA SET nha\_ttrang = N'Đã bán' WHERE nha\_id = @MaNha | **X(NHA)**  Xin khoá ghi trên bảng NHA để sửa lại tình trạng nhà |
|  |  | COMMIT |  |
| update NHA  set  nha\_slphong=@nha\_slphong,  nha\_ttrang=@nha\_ttrang,  nha\_duong=@nha\_duong,  nha\_quan=@nha\_quan,  nha\_kvuc=@nha\_kvuc, nha\_tp=@nha\_tp,  nha\_giat=@nha\_giat,  nha\_giab=@nha\_giab,  nha\_dkban=@nha\_dkban,  nha\_idloai=@nha\_idloai,  nha\_idchu=@nha\_idchu,  nha\_idcn=@nha\_idcn,  nha\_idnv=@nha\_idnv,  nha\_ndang=@nha\_ndang,  nha\_hethan=@nha\_hethan  where nha\_id=@nha\_id | **X(NHA)**  Xin khoá ghi trên bảng NHA để cập nhật lại các thông tin nhà |  |  |
| COMMIT |  |  |  |

## **Giải thích xử lý tranh chấp**: Bước đầu tiên Admin tiến hành cập nhật thông tin nhà cho 1 căn nhà. Trong quá trình đó, một nhân viên tiến hành lập hợp đồng cho căn nhà mà Admin đang cập nhật thông tin, sau khi lập xong hợp đồng thì nhân viên cập nhật lại tình trạng nhà là đã cho thuê/đã bán cho căn nhà đó. Vì ra đã set mức cô lập REAPEATABLE READ cho giao tác 1 nên sau khi Admin tiến hành cập nhật thông tin nhà xong (đã commit) thì nhân viên lập hợp đồng mới cập nhật được tình trạng nhà đã cho thuê/đã bán 🡪 Không bị mất dữ liệu cập nhật của nhân viên lập hợp đồng 🡪 Xử lý được lỗi Lost update.

## Tình huống 2: Admin đang tiến hành thêm 1 khách hàng mới. Cùng lúc đó, nhân viên tiến hành xem thông tin của 1 khách hàng

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR02: Dirty read**  T1 (User = admin): thực hiện việc thêm khách hàng mới  T2 (User = nhân viên): thực hiện xem thông tin khách hàng | | | |
| **Ttkh\_\_insert** | **Khóa** | **NV\_XemThongTinKhachHang** | **Khóa** |
| ***Input***: Các thuộc tính thông tin của khách hàng  ***Output***: Khách hàng được thêm vào bảng KHACH\_HANG nếu được commit, ngược lại sẽ rollback nếu dữ liệu bị lỗi. | ***Input:*** Mã khách hàng  ***Output***: Các thông tin của khách hàng |
| SET TRANSACTION ISOLATION  LEVEL **READ COMMITED** | SET TRANSACTION ISOLATION  LEVEL **READ COMMITED** |
| BEGIN TRAN |  |  |  |
| BEGIN TRY |  |  |  |
| B1: Kiểm tra mã khách hàng có tồn tại hay không  if exists(select \* from KHACH\_HANG where kh\_id=@kh\_id)  begin  print N'Khach hang' + cast(@kh\_id as varchar(3)) + N' Đã Tồn Tại'  rollback tran  return 1  end | **R(KHACH\_HANG)**  Xin khóa đọc trên bảng KHACH\_HANG với đk  kh\_id = @MaKH |  |  |
| B2: Kiểm tra tên khách hàng có hợp lệ hay không  if @kh\_ten is null or @kh\_ten = ''  begin  print N'Tên Khach Hang Không Được Trống'  rollback tran  return 1  end |  |  |  |
| B3: Kiểm tra SDT khách hàng có hợp lệ hay không  IF LEN(@kh\_sdt) <> 10  begin  print N'SDT Không Hợp Lệ'  rollback tran  return 1  end |  |  |  |
| B4: Kiểm tra địa chỉ có hợp lệ hay không  if @kh\_dchi is null or @kh\_ten = ''  begin  print N'Dia Khach Hang Không Được Trống'  rollback tran  return 1  end |  |  |  |
| B5: Kiểm tra chi nhánh có hợp lệ hay không  if not exists(select \* from CHI\_NHANH where cn\_id=@kh\_idcn)  begin  print N'Chi Nhánh '+ @kh\_idcn + N' Không Tồn Tại'  rollback tran  return 1  end | **R(CHI\_NHANH)**  Xin khóa đọc trên bảng CHI\_NHANH với đk  cn\_id=@kh\_idcn |  |  |
| B6: Thêm thông tin khách hàng mới vào bảng KHACH\_HANG  insert into KHACH\_HANG (kh\_id,kh\_ten,kh\_dchi,kh\_sdt,kh\_idcn)  values (@kh\_id,@kh\_ten,@kh\_dchi,@kh\_sdt,@kh\_idcn) | **X(KHACH\_HANG)**  Xin khoá ghi trên bảng KHACH\_HANG để thêm khách hàng mới  Giữ khóa cho đến hết giao tác |  |  |
| END TRY |  |  |  |
| WAITFOR DELAY '00:00:05' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: Kiểm tra khách hàng có tồn tại hay không  IF NOT EXISTS(SELECT \* FROM KHACH\_HANG WHERE kh\_id = @MaKH)  BEGIN  PRINT 'KH' + CAST(@MaKH AS VARCHAR(3)) + N' Không Tồn Tại'  ROLLBACK TRAN  RETURN  END | **R(KHACH\_HANG)**  Xin khóa đọc trên bảng KHACH\_HANG với đk  kh\_id = @MaKH  (Do Tran1 đang giữ khóa trên bảng KHACHHANG nên chờ đến khi Tran1 thực hiện xong) |
|  |  | B2: Hiển thị thông tin khách hàng có mã đã nhập  SELECT kh\_id 'Mã khách hàng',kh\_ten 'Tên Khách Hàng',kh\_sdt 'SĐT',kh\_dchi 'Địa Chỉ'  FROM KHACH\_HANG  WHERE kh\_id = @MaKH | **R(KHACH\_HANG)**  Xin khóa đọc trên bảng KHACH\_HANG với đk  kh\_id = @MaKH |
|  |  | COMMIT |  |
| ROLLBACK TRAN //khách hàng mới trùng SĐT với khách hàng đã có trong bảng (ràng buộc SĐT của khách hàng là duy nhất)  return 1 |  |  |  |
| BEGIN CATCH |  |  |  |
| print N'LỖI HỆ THỐNG'  ROLLBACK TRAN  return 1 |  |  |  |
| END CATCH |  |  |  |
| COMMIT |  |  |  |

## **Giải thích xử lý tranh chấp**: Bước đầu tiên Admin tiến hành thêm 1 khách hàng mới vào bảng KHACH\_HANG, khách hàng được thêm vào bảng KHACH\_HANG. Trong lúc đó, nhân viên tiến hành xem thông tin khách hàng. Tuy nhiên sau khi Insert thì khách hàng mới vi phạm ràng buộc (trùng SĐT với khách hàng có trong bảng KHACH\_HANG) dẫn đến Rollback toàn bộ dữ liệu của khách hàng mới được thêm vào bởi Admin. Do đã set mức cô lập READ COMMITED cho giao tác 2 nên nó phải đợi giao tác 1 thực hiện hết giao tác mới được thực hiện đọc trên bảng KHACH\_HANG 🡪 Xử lý được lỗi Dirty Read.

# Sinh viên thực hiện: QUÁCH PHÚ THÀNH - 18127215

## Tình huống 1: Conversion Deadlock

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: Conversion Deadlock**  T1 (User = nhân viên): thực hiện xóa một khách hành có id A  T2 (User = quản lí): thực hiện xóa một khách hành có id A | | | |
| ttkh\_delete | **Khóa** | ttkh\_delete2 | **Khóa** |
| ***Input:*** Id khách hàng  ***Output***: Bảng khách hàng không còn khách hàng có id đó nữa | ***Input:*** Id khách hàng  ***Output***: Thông báo khách hàng không tồn tại |
| SET TRANSACTION ISOLATION  LEVEL SERIALIZABLE | SET TRANSACTION ISOLATION  LEVEL SERIALIZABLE |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra thông tin: Xem id khách hàng có tồn tại hay không  begin tran  set tran isolation level serializable  if not exists (select \*from KHACH\_HANG with (Updlock) where kh\_id=@kh\_id)  begin  print N'Khach Hang' + cast(@kh\_id as varchar(3)) + N'Khong ton tai'  rollback tran  return 1  end | **R1(A)**  //Xin khoá đọc trên A và giữ lại đến hết giao tác  (Update lock) |  |  |
| WAITFOR DELAY '00:00:15' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: Kiểm tra thông tin: Xem id khách hàng có tồn tại hay không  begin tran  set tran isolation level serializable  if not exists (select \*from KHACH\_HANG with (Updlock) where kh\_id=@kh\_id)  begin  print N'Khach Hang' + cast(@kh\_id as varchar(3)) + N'Khong ton tai'  rollback tran  return 1  end | **R2(A)**  //Chờ T1 (Vì uplock không tương thích với uplock) |
| begin try  delete from KHACH\_HANG where kh\_id=@kh\_id  end try  begin catch  DECLARE @ErrorMsg VARCHAR(2000)  SELECT @ErrorMsg = N'Lỗi: ' + ERROR\_MESSAGE()  RAISERROR(@ErrorMsg, 16,1)  ROLLBACK TRAN  return 1  end catch  commit tran  return 0 | **W1(A)**  //Khóa update được năng thành khóa X |  |  |
|  |  | WAITFOR DELAY '00:00:15' |  |
|  |  | begin try  delete from KHACH\_HANG where kh\_id=@kh\_id  end try  begin catch  DECLARE @ErrorMsg VARCHAR(2000)  SELECT @ErrorMsg = N'Lỗi: ' + ERROR\_MESSAGE()  RAISERROR(@ErrorMsg, 16,1)  ROLLBACK TRAN  return 1  end catch  commit tran  return 0 | **W2(A)** |

**Giải thích xử lý tranh chấp:** do T1 giữ khóa update trên khách hàng có id A (đang ở mức serializable) đến khi T1 ghi thì trở thành khóa X. Lúc T1 vừa xin được khóa update thì T2 cũng xin khóa update trên A 🡪 Không xin được khóa vì T1 đang giữ khóa update trên A 🡪 Chờ đến khi T1 commit. Sau khi T1 commit (đã xóa khách hàng có id A), thì T2 mới vào xin khóa update trên A, nhưng A hiện đã không tồn tại nên gt bị rollback (đây là kết quả mong muốn).

## Tình huống 2: Cycle Deadlock

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: Cycle Deadlock**  T1 (User = nhân viên): thực hiện update nhận xét của khách hàng vể một căn nhà A  T2 (User = nhân viên): thực hiện update số lượt xem của nhà A | | | |
| upd\_nxet | **Khóa** | CapNhatLuotXem | **Khóa** |
| ***Input:*** Id khách hàng, id nhà, nhận xét  ***Output***: Dòng nhận xét ở bảng XEM\_NHA được cập nhật | ***Input:*** Id khách hàng, id nhà  ***Output***: Cột số lượt xem được cập nhật |
| SET TRANSACTION ISOLATION  LEVEL SERIALIZABLE | SET TRANSACTION ISOLATION  LEVEL SERIALIZABLE |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra thông tin: Xem nhà có tồn tại và còn hạn để sửa đổi hay không  begin tran  declare @check date = (select n.nha\_hethan from NHA n where n.nha\_id = @idnha);  if (@check < GETDATE() or @check = null)  begin  raiserror (N'Đã hết hạn đăng của nhà hoặc nhà không tồn tại', 16, 0)  rollback  return 1  end | **R1(NHA)**  //Xin khoá đọc trên NHA và giữ lại đến hết giao tác |  |  |
| WAITFOR DELAY '00:00:15' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: Kiểm tra thông tin: Xem lượt xem nhà của căn nhà đó có tồn tại dòng nào hay không  if exists (select \* from XEM\_NHA xn with(nolock) where xn.x\_idkh = @idkh and xn.x\_idnha = @idnha) | **R2(XEM\_NHA)**  //Không xin khóa |
|  |  | B2: Đếm số dòng có id nhà A  begin  declare @dem int = (select COUNT(x\_idnha) from XEM\_NHA xn with(nolock) where xn.x\_idnha = @idnha) | **R2(XEM\_NHA)**  // Không xin khóa |
|  |  | B3: Cập nhật lượt xem ở bảng NHA dựa trên việc có tồn tại bao nhiêu lần xem ở XEM\_NHA  UPDATE NHA SET nha\_luotxem = @dem where nha\_id = @idnha  end  else  UPDATE NHA SET nha\_luotxem = 0 where nha\_id = @idnha | **W2(NHA)**  //Chờ T1 commit để xin khóa ghi trên NHA do T1 đang giữ khóa đọc trên NHA |
| B2: Cập nhật nhận xét vào đúng dòng có id nhà và id kh ở bảng XEM\_NHA  begin try  UPDATE XEM\_NHA SET x\_nxet = @nxet WHERE x\_idnha = @idnha and x\_idkh = @idkh  end try  begin catch  DECLARE @ErrorMsg VARCHAR(2000)  SELECT @ErrorMsg = N'Lỗi: ' + ERROR\_MESSAGE()  RAISERROR(@ErrorMsg, 16,1)  ROLLBACK TRAN  return 1  end catch  commit tran  return 0 | **W1(XEM\_NHA)**  //Xin khóa ghi trên XEM\_NHA, ghi dữ liệu và commit 🡪 Trả khóa trên XEM\_NHA và NHA |  |  |

**Giải thích xử lý tranh chấp:** do T1 giữ khóa đọc S trên bảng NHA có id nhà A (đang ở mức serializable) đến hết giao tác và T2 không giữ khóa đọc S trên bảng XEM\_NHA có id nhà A nên thời điểm sau đó T2 không xin được khóa X trên NHA vì T1 đang giữ khóa S, nhưng T1 xin được khóa X trên XEM\_NHA vì T2 không giữ khóa 🡪 T1 xin được khóa X và thực thi 🡪 T1 commit và trả khóa trên NHA 🡪 T2 xin được khóa và thực hiện tiếp đến hết. Ở đây em sử dụng Nolock trên 2 lần đọc của T2 vì nó mang tính chất đếm nên cho dù có chênh lệch (có thể có giao tác khác thêm mới) cũng không sao, không gây ra lỗi quá nghiêm trọng cho cả database và T2 chỉ cập nhật lượt xem nên có thể chạy lại sau để cập nhật tiếp (lượt xem xuất ra cho khách hàng xem không cần phải quá chính xác).