**d**

BÁO CÁO ĐỒ ÁN THỰC HÀNH

MÔN HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU

GVHD: ThS. Hồ Thị Hoàng Vy,   
ThS. Tiết Gia Hồng

Nhóm: 23.CQ.HQTCSDL.18

Đợt 4

# **THÔNG TIN NHÓM**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **MSSV** | **Họ tên** | **Công việc** | **% Hoàn thành** |
| **1** | 21120289 | Diệp Quốc Hoàng Nam | Xử lý tình huống tranh chấp: Dirty Read, Phantom Read | 100% |
| **2** | 21120538 | Lưu Vĩnh Quang | Xử lý tình huống tranh chấp: Lost update, Dirty read | 100% |
| **3** | 21120548 | Lâm Chanh Cô Sol | Xử lý tình huống tranh chấp: Dirty read, Phantom Read | 100% |
| **4** | 21120551 | Đàm Tử Tâm | Xử lý tình huống tranh chấp: Dirty Read, Unrepeatable read | 100% |
| **5** | 21120555 | Nguyễn Hữu Thắng | Xử lý tình huống tranh chấp: Dirty Read, Unrepeatable read | 100% |
| **6** | 21120582 | Đinh Hoàng Trung | Xử lý tình huống tranh chấp: Lost Update, Phantom Read | 100% |

# XỬ LÍ TÌNH HUỐNG TRANH CHẤP

# Sinh viên thực hiện: 21120289 – Diệp Quốc Hoàng Nam

## Tình huống 1: Dirty read



|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: *Dirty read***  T1 (User = quản trị viên): thực hiên cập nhật thông tin chi tiết của thuốc  T2 (User = quản trị viên): thực hiện xem thông tin chi tiết của thuốc | | | |
| **sp\_SuaThuoc** | **Khóa** | **sp\_XemThuoc**  **CoDieuKien** | **Khóa** |
| ***Input:*** @MaThuoc, @TenThuoc, @DonViTinh, @ChiDinh, @SoLuongTon, @NgayHetHan  ***Output***: Thông tin của thuốc có @MaThuoc tương ứng sẽ được cập nhật, nếu tồn tại một tham số truyền vào không hợp lệ hoặc sai về ý nghĩa thì giao tác sẽ bị hủy. | ***Input***: @MaThuoc  ***Output***: Chi tiết của một thuốc bao gồm các thuộc tính tương ứng, nếu không tồn tại thì giao tác không được thực hiện |
| SET TRANSACTION ISOLATION  LEVEL **READ COMMITED** | SET TRANSACTION ISOLATION  LEVEL **READ COMMITED** |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra mã vạch được truyền vào có tồn tại không.  IF NOT EXISTS (SELECT \* FROM THUOC WHERE MaThuoc = @MaThuoc)  BEGIN  PRINT (N ‘Mã thuốc không tồn tại’)  ROLLBACK TRAN  RETURN 1  END | **R(THUOC)**  //Xin khóa đọc trên bảng THUOC |  |  |
| B2: Thay đổi thông tin thuốc có mã tương ứng  UPDATE THUOC  SET TenThuoc = @TenThuoc, DonViTinh = @DonViTinh, ChiDinh = @ChiDinh, NgayHetHan = @NgayHetHan, SoLuongTon = @SoLuongTon  WHERE MaThuoc = @MaThuoc | **X(THUOC)**  //Xin khóa ghi trên bảng THUOC, khóa này sẽ được giữ cho đến hết giao tác. |  |  |
| WAITFOR DELAY '00:00:05' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: Kiểm tra thông tin mã thuốc có tồn tại  IF NOT EXISTS (SELECT \* FROM THUOC WHERE MaThuoc = @MaThuoc)  BEGIN  PRINT (N ‘Mã thuốc không tồn tại’)  ROLLBACK TRAN  RETURN 1  END | **R(THUOC)**  //Xin khóa đọc trên bảng thuốc, do mức cô lập là read commited, không được cấp khóa vì phải chờ giao tác giữ khóa phải trả khóa ghi đang còn trên bảng THUOC |
|  |  | B2: Xuất danh mục thuốc  SELECT \* FROM THUOC  WHERE MaThuoc = @MaThuoc | **R(THUOC)**  //Xin khóa đọc trên bảng thuốc, do mức cô lập là read commited, không được cấp khóa vì phải chờ giao tác giữ khóa phải trả khóa ghi đang còn trên bảng THUOC |
|  |  | COMMIT |  |
| B3: Kiểm tra ngày hết hạn có hợp lệ  IF DATEDIFF(GETDATE(),@NgayHetHan,DD) <= 0  BEGIN  PRINT(N ‘Ngày hết hạn phải sau ngày hiện tại’)  ROLLBACK TRAN  RETURN 1  END |  |  |  |
| COMMIT |  |  |  |

Nét màu đỏ: Thời gian khóa được giữ Nét màu vàng: Đường đi mong muốn Nét màu xanh: Đường đi thật sự.

## Tình huống 2: Phantom read



|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: *Phantom read***  T1 (User = nha sĩ): thực hiên kê đơn thuốc  T2 (User = quản trị viên): thực hiện đăng kí lịch khám cho bệnh nhân | | | |
| **sp\_KeDonThuoc** | **Khóa** | **sp\_ThemThuoc** | **Khóa** |
| ***Input:*** Mã Thuốc (@MaThuoc), số điện thoại khách hàng (@sdt), Mã bệnh án (@MaBA) mã bệnh án phân biệt các bệnh án của 1 khách hàng, Số lượng kê đơn (@soluong)  ***Output***: Thông báo thêm chi tiết đơn thuốc thành công, in ra đơn thuốc, bên cạnh đó thì số lượng thuốc tồn kho sẽ được cập nhật lại ngay sau khi chi tiết đơn thuốc được thêm thành công. Ngược lại, thông báo thất bại nếu thêm khi thành công. | ***Input:*** @MaThuoc, @TenThuoc, @DonViTinh, @ChiDinh, @SoLuongTon, @NgayHetHan  ***Output***: Thông tin của thuốc có @MaThuoc tương ứng sẽ được thêm vào, nếu tồn tại một tham số truyền vào không hợp lệ hoặc sai về ý nghĩa thì giao tác sẽ bị hủy. |
| SET TRANSACTION ISOLATION  LEVEL **READ COMMITED** | SET TRANSACTION ISOLATION  LEVEL **READ COMMITED** |
| BEGIN TRAN |  |  |  |
| B0: Đưa ra các phương án về thuốc cho nha sĩ  select \* from THUOC | **S(THUOC)**  Xin khóa S trên bảng thuốc, khóa này sẽ được giữ cho tới khi đọc xong |  |  |
| B1: Kiểm tra thông tin MaThuoc tồn tại hay không  if (not exists (select \* from THUOC where MaThuoc=@MaThuoc))  begin  print N'Thuốc này không tồn tại'  rollback tran  return 1  end | **S(THUOC)**  Xin khóa S trên bảng thuốc, khóa này sẽ được giữ cho tới khi đọc xong |  |  |
| B1: Kiểm tra thông tin (1) ChiTietDonThuoc đã được kê vào bệnh án hay chưa.  if exists(select \*  from CHITIETDONTHUOC  where MaThuoc = @MaThuoc and MaBenhAn = @MaBA and SdtKhachHang = @sdt)  begin  print N'Chi tiết đơn thuốc đã tồn tại - Thuốc đã được kê trong bệnh án ' + @MaBA + N' của khách hàng có số điện thoại ' + @sdt  rollback tran  return 1  end | **S(CHITIET DONTHUOC)**  Xin khóa S trên bảng chi tiết đơn thuốc, khóa này sẽ được giữ tới khi đọc xong. |  |  |
| B2: Kiểm tra thông tin (2) KhachHang có tồn tại hay không.  if not exists (select \* from KHACHHANG where Sdt = @sdt)  begin  print N'Không tồn tại khách hàng có số điện thoại: ' + @sdt  rollback tran  return 1  end | **S(KHACHHAN G)** Xin khóa S trên bảng khách hàng, khóa này sẽ được giữ tới khi đọc xong |  |  |
| WAITFOR DELAY '00:00:05' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | IF(EXISTS(SELECT \* FROM THUOC WHERE MaThuoc = @MaThuoc)  BEGIN  ROLLBACK TRAN  RETURN 1  END | **S(THUOC)**  Xin khóa S trên thuốc, khóa này sẽ được giữ cho tới khi đọc xong |
|  |  | INSERT INTO THUOC VALUES (@MaThuoc, @TenThuoc, @DonViTinh, @ChiDinh, @SoLuongTon, @NgayHetHan) | **X(THUOC)**  Xin khóa X trên bảng thuốc, khóa được giữ cho tới hết giao tác. |
|  |  | IF DATEDIFF(DD,@NgayHetHan,GETDATE()) > 0  BEGIN  PRINT N'Ngày hết hạn phải sau ngày hiện tại'  ROLLBACK TRAN  RETURN 1  END |  |
| B3: Kiểm tra thông tin (3) Kiểm tra xem còn đủ thuốc để kê đơn hay không.  if @soluong > (select soluongtonkho from THUOC where MaThuoc = @MaThuoc)  begin  print N'Không còn đủ thuốc ' + @Mathuoc + N' để thực hiện kê đơn.'  rollback tran  return 1  end | **S(THUOC)**  Xin khóa S trên bảng thuốc, khóa này sẽ được giữ cho tới khi đọc xong |  |  |
| update THUOC  set SoLuongTonKho = SoLuongTonKho - @soluong  where MaThuoc = @MaThuoc | **X(THUOC)**  Xin khóa X trên bảng thuốc, khóa được giữ cho tới hết giao tác. |  |  |
| insert into CHITIETDONTHUOC  values (@MaThuoc, @sdt, @MaBA, @soluong) | **S(CHITIET DONTHUOC)**  Xin khóa X trên bảng chi tiết đơn thuốc, khóa được giữ cho tới hết giao tác.. |  |  |
|  |  | COMMIT |  |
| COMMIT |  |  |  |



Nét màu đỏ: Thời gian khóa được giữ Nét màu vàng: Đường đi mong muốn Nét màu xanh: Đường đi thật sự.

# Sinh viên thực hiện: 21120538 – Lưu Vĩnh Quang

## Tình huống 1: Lost update

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: Lost update**  T1 (User = quản lý): thực hiện thêm 1 thông tin chuyến xe vào ngày A.  T2 (User = tổ trưởng): thực hiện thống kê số chuyến xe sẽ thực hiện trong ngày A | | | |
| **sp\_ThemChiTietDonThuoc1** | **Khóa** | **Sp\_ThemChiTietDonThuoc2** | **Khóa** |
| ***Input:*** *@mathuoc,@sdt,@maba,@sl*  ***Output***: Thông tin chi tiết của hóa đơn được cập nhập. Số lượng tồn của thuốc cũng được cập nhập theo | ***Input:*** *@mathuoc,@sdt,@maba,@sl*  ***Output***: Thông tin chi tiết của hóa đơn được cập nhập. Số lượng tồn của thuốc cũng được cập nhập theo |
| SET TRANSACTION ISOLATION  LEVEL **READ COMMITED** | SET TRANSACTION ISOLATION  LEVEL **READ COMMITED** |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra thông tin xem mã thuốc đã tồn tại trong bảng THUOC hay chưa  if not exists (select \* from THUOC where THUOC.MaThuoc = @mathuoc)  begin  PRINT CAST(@mathuoc AS VARCHAR(50)) + N' Không Tồn Tại'  rollback tran  return 1  end | **R(THUOC)**  Xin khóa đọc trên bảng THUOC |  |  |
| B2: Kiểm tra thông tin khách hàng và bệnh án  if not exists (select \* from KHACHHANG where KHACHHANG.Sdt = @sdt)  begin  PRINT N'Khách Hàng Không Tồn Tại'  rollback tran  return 1  end  if not exists (select \* from BENHAN where BENHAN.MaBenhAn = @maba)  begin  PRINT N'Bệnh Án Không Tồn Tại'  rollback tran  return 1  end | **R(KHACHHANG)**  Xin khóa đọc trên bang KHACHHANG  **R(BENHAN)**  Xin khóa đọc trên bảng BENHAN |  |  |
| B3: Đọc số lượng tồn của thuốc (có cấp khóa update trên dòng), kiểm tra xem còn đủ thuốc để cấp không  declare @slton int  set @slton = (select SoLuongTonKho from THUOC with (ROWLOCK,UPDLOCK) where THUOC.MaThuoc = @mathuoc)  if (@slton < @sl)  begin  PRINT N'Số lượng thuốc tồn kho không còn đủ'  rollback tran  return 1  end | **X(THUOC(row))**  Xin khóa ghi trên dòng được chọn, giữ đến hết giao tác |  |  |
| WAITFOR DELAY '00:00:05' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: Kiểm tra thông tin xem mã thuốc đã tồn tại trong bảng THUOC hay chưa  if not exists (select \* from THUOC where THUOC.MaThuoc = @mathuoc)  begin  PRINT CAST(@mathuoc AS VARCHAR(50)) + N' Không Tồn Tại'  rollback tran  return 1  end | **R(THUOC)**  Xin khóa đọc, trả ngay sau khi đọc xong |
|  |  | B2: Kiểm tra thông tin khách hàng và bệnh án  if not exists (select \* from KHACHHANG where KHACHHANG.Sdt = @sdt)  begin  PRINT N'Khách Hàng Không Tồn Tại'  rollback tran  return 1  end  if not exists (select \* from BENHAN where BENHAN.MaBenhAn = @maba)  begin  PRINT N'Bệnh Án Không Tồn Tại'  rollback tran  return 1  end | **R(KHACHHANG)**  Xin khóa đọc, trả ngay sau khi đọc xong  **R(BENHAN)**  Xin khóa đọc, trả ngay sau khi đọc xong |
|  |  | B3: Đọc số lượng tồn của thuốc (có cấp khóa update trên dòng), kiểm tra xem còn đủ thuốc để cấp không  declare @slton int  set @slton = (select SoLuongTonKho from THUOC WITH (ROWLOCK,UPDLOCK) where THUOC.MaThuoc = @mathuoc)  if (@slton < @sl)  begin  PRINT N'Số lượng thuốc tồn kho không còn đủ'  rollback tran  return 1  end | **X(THUOC(row))**  Xin khóa ghi trên dòng được chọn, giữ đến hết giao tác (Không xin được do trên dòng được chọn đang có khóa Update) |
|  |  | B4: Cập nhập số lượng tồn  set @slton = @slton - @sl  update THUOC  set SoLuongTonKho = @slton  where MaThuoc = @mathuoc  insert into CHITIETDONTHUOC values (@mathuoc,@sdt,@maba,@sl) |  |
| B4: Cập nhập số lượng tồn  set @slton = @slton - @sl  update THUOC  set SoLuongTonKho = @slton  where MaThuoc = @mathuoc  insert into CHITIETDONTHUOC values (@mathuoc,@sdt,@maba,@sl) |  |  |  |
| IF @error ….  ROLLBACK |  | COMMIT |  |



## Tình huống 2: DIRTY READ



|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: *Dirty read***  T1 (User = Nha sĩ): thực hiện thêm 1 chi tiết đơn thuốc vào bệnh án của khách hàng.  T2 (User = Khách hàng): thực hiện xem thông tin đơn thuốc của bệnh án | | | |
| **sp\_ThemChiTietDonThuoc** | **Khóa** | **Sp\_XemThongTinBenhAn\_DonThuoc** | **Khóa** |
| ***Input:*** *@mathuoc,@sdt,@maba*  ***Output***: Thông tin chi tiết đơn thuốc của bệnh án được cập nhập | ***Input***: @sdt,@maba  ***Output***: Thông tin đơn thuốc của bệnh án của khách hàng |
| SET TRANSACTION ISOLATION  LEVEL **READ COMMITTED** | SET TRANSACTION ISOLATION  LEVEL **READ COMMITTED** |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra thông tin xem mã thuốc có tồn tại hay không  if not exists (select \* from THUOC where THUOC.MaThuoc = @mathuoc)  begin  PRINT CAST(@mathuoc AS VARCHAR(50)) + N' Không Tồn Tại'  rollback tran  return 1  end | **R(THUOC)**  Xin khóa đọc trên bảng THUOC, trả lại ngay sau khi đọc xong |  |  |
| B2: Kiểm tra thông tin xem sđt khách hàng có tồn tại hay không  if not exists (select \* from KHACHHANG where KHACHHANG.Sdt = @sdt)  begin  PRINT N'Khách Hàng Không Tồn Tại'  rollback tran  return 1  end | **R(ChuyenDi)**  Xin khoá đọc trên bảng KHACHHANG, trả lại ngay sau khi đọc xong |  |  |
| B3: Kiểm tra xem mã bệnh án có tồn tại hay không  if not exists (select \* from BENHAN where BENHAN.MaBenhAn = @maba)  begin  PRINT N'Bệnh Án Không Tồn Tại'  rollback tran  return 1  end | **R(BENHAN)**  Xin khóa đọc trên bảng BENHAN, trả lại ngay sau khi đọc xong |  |  |
| B4: Thêm thông tin vào bảng CHITIETDONTHUOC  declare @slton int  set @slton = (select SoLuongTonKho from THUOC where THUOC.MaThuoc = @mathuoc)  set @slton = @slton - @sl  update THUOC  set SoLuongTonKho = @slton  where MaThuoc = @mathuoc  insert into CHITIETDONTHUOC values (@mathuoc,@sdt,@maba,@sl) | **X(CHITIETDONTHUOC)**  Xin khóa ghi trên bảng CHITIETDONTHUOC, giữ đến hết giao tác |  |  |
| WAITFOR DELAY '00:00:05' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: Kiểm tra xem bệnh án có tồn tại hay không  if not exists (  select \* from BENHAN  where MaBenhAn = @maba and SdtKhachHang = @sdt  )  begin  PRINT N'Bệnhh án không Tồn Tại'  rollback tran  return 1  end | **R(BENHAN)**  Xin khóa đọc trên bảng BENHAN, trả lại ngay sau khi đọc xong |
|  |  | B2: Thống kê đơn thuốc của bệnh án  select \* from BENHAN BA, CHITIETDONTHUOC  where BA.MaBenhAn = @maba and BA.SdtKhachHang = @sdt and CHITIETDONTHUOC.MaBenhAn = BA.MaBenhAn | **R(CHITIETDONTHUOC)**  //Xin khóa đọc trên CHITIETDONTHUOC, trả lại ngay sau khi đọc xong (Không xin được do trên bảng CHITIETDONTHUOC đang có khóa X) |
| IF @error ….  //Khi insert đến bảng A: dữ liệu bị lỗi, không thể insert vào bảng A được. Điều này dẫn đến toàn dữ liệu sẽ bị ROLLBACK  //Gợi ý: Khi exec store, đưa ra dữ liệu làm sao cho insert bảng A bị lỗi: như trùng khoá chính, hay bị đụng ràng buộc trigger.  //Phải đưa ra được tình huống ROLLBACK thực tế, chứ không phải muốn cho ROLLBACK tuỳ ý.  ROLLBACK |  | COMMIT |  |



# Sinh viên thực hiện: 21120548 – Lâm Chanh Cô Sol

## Tình huống 1: Dirty read

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: *Dirty read***  T1 (User = Nha sĩ): ghi lại hồ sơ bệnh nhân.  T2 (User = Nhân viên): Tra cứu hồ sơ khám bệnh của bệnh nhân. | | | |
| **sp\_GhiHoSoBenhNhan** | **Khóa** | **sp\_TraCuuHoSoBenhNhan** | **Khóa** |
| ***Input:*** Số điện thoại khách hàng (@SdtKhachHang), mã bệnh án (@MaBenhAn), mã thuốc (@MaThuoc), số lượng thuốc (@SoLuong)  ***Output***: Cập nhật lại số lượng thuốc của @MaThuoc tương ứng trong chi tiết đơn thuốc của hồ sơ bệnh nhân có số điện thoại là @SdtKhachHang và mã bệnh án là @MaBenhAn.  Nhưng cập nhật sẽ không được thực hiện do số lượng thuốc cập nhật lớn hơn số lượng thuốc đang tồn kho. | ***Input***: Số điện thoại khách hàng (@SdtKhachHang), mã bệnh án (@MaBenhAn)  ***Output***: Chi tiết đơn thuốc trong hồ sơ bệnh án của bệnh nhân sẽ hiện nội dung không chính xác về số lượng thuốc của đơn thuốc được chỉnh sửa của giao tác ghi lại hồ sơ bệnh nhân bên kia. |
| SET TRANSACTION ISOLATION  LEVEL **READ COMMITTED** | SET TRANSACTION ISOLATION  LEVEL **READ COMMITTED** |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra thông tin @SdtKhachHang, @MaBenhAn có tồn tại không  IF NOT EXISTS (SELECT \* FROM BENHAN  WHERE SdtKhachHang=@SdtKhachHang AND MaBenhAn = @MaBenhAn)  BEGIN  PRINT(N’Hồ sơ bệnh án không tồn tại’)  ROLLBACK TRAN  RETURN 1  END | **R(BENHAN)**  //Xin khoá đọc trên bảng BENHAN |  |  |
| B2: Kiểm tra thông tin @MaThuoc có tồn tại không  IF NOT EXISTS (SELECT \* FROM THUOC WHERE MaThuoc=@MaThuoc)  BEGIN  PRINT(N’Thuốc không tồn tại trong kho’)  ROLLBACK TRAN  RETURN 1  END | **R(THUOC)**  //Xin khoá đọc trên bảng THUOC |  |  |
| B3: Kiểm tra thông tin @SdtKhachHang, @MaBenhAn, @MaThuoc có tồn tại trong bảng CHITIETDONTHUOC không  IF NOT EXISTS (SELECT \* FROM CHITIETDONTHUOC WHERE  SdtKhachHang = @SdtKhachHang AND  MaBenhAn = @MaBenhAn AND  MaThuoc=@MaThuoc)  BEGIN  PRINT(N’Chi tiết đơn thuốc này không tồn tại’)  ROLLBACK TRAN  RETURN 1  END | **R(CHITIETDONTHUOC)**  //Xin khoá đọc trên bảng CHITIETDONTHUOC |  |  |
| B4: Chỉnh sửa thông tin về số lượng thuốc của @MaThuoc trong chi tiết đơn thuốc của bệnh án có khóa là @SdtKhachHang và @MaBenhAn  UPDATE CHITIETDONTHUOC  SET SoLuong = @SoLuong  WHERE SdtKhachHang = @SdtKhachHang  AND MaBenhAn = @MaBenhAn  AND MaThuoc = @MaThuoc | **X(CHITIETDONTHUOC)**  //Xin khóa ghi trên bảng CHITIETDONTHUOC , khóa này sẽ được giữ cho đến hết giao tác |  |  |
| WAITFOR DELAY '00:00:20' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: Kiểm tra thông tin @SdtKhachHang, @MaBenhAn có tồn tại không  IF NOT EXISTS (SELECT \* FROM BENHAN  WHERE SdtKhachHang=@SdtKhachHang AND MaBenhAn = @MaBenhAn)  BEGIN  PRINT(N’Hồ sơ bệnh án không tồn tại’)  ROLLBACK TRAN  RETURN 1  END | **R(BENHAN)**  //Xin khóa đọc trên bảng BENHAN |
|  |  | B2: Xuất ra hồ sơ bệnh nhân gồm: số điện thoại, họ tên, thời gian khám (ngày, giờ), dịch vụ và đơn thuốc  SELECT BA.SdtKhachHang, KH.HoTen, PH.Ngay, LT.GioBatDau, DV.TenDichVu, T.TenThuoc, CTDT.SoLuong  FROM KHACHHANG KH, BENHAN BA, PHIEUHEN PH, LICHTRUC LT, CHITIETDICHVU CTDV, DICHVU DV, CHITIETDONTHUOC CTDT, THUOC T  WHERE BA.SdtKhachHang = KH.Sdt AND PH.SdtKhachHang = BA.SdtKhachHang  AND LT.Ngay = PH.Ngay  AND LT.ThuTuCa = PH.ThuTuCa  AND CTDV.SdtKhachHang = BA.SdtKhachHang  AND CTDV.MaBenhAn = BA.MaBenhAn AND CTDV.MaDichVu = DV.MaDichVu AND CTDT.SdtKhachHang = BA.SdtKhachHang  AND CTDT.MaBenhAn = BA.MaBenhAn AND CTDT.MaThuoc = T.MaThuoc | **R(KHACHHANG, BENHAN, PHIEUHEN, LICHTRUC, CHITIETDICHVU, DICHVU, CHITIETDONTHUOC, THUOC)**  //Xin khóa đọc trên bảng các bảng để xuất ra đầy đủ thông tin về hồ sơ bệnh nhân |
|  |  | COMMIT |  |
| B5: Kiểm tra số lượng thuốc có hợp lệ không  IF @SoLuong > (SELECT SoLuongTonKho FROM THUOC  WHERE MaThuoc = @MaThuoc)  BEGIN  PRINT (N’Số lượng thuốc trong kho không đủ’)  ROLLBACK  RETURN 1  END |  |  |  |
| COMMIT |  |  |  |

## Tình huống 2: Phantom read

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR02: Phantom read**  T1 (User = Nha sĩ): xem lịch hẹn.  T2 (User = Nhân viên): xóa lịch hẹn. | | | |
| **sp\_XemLichHen** | **Khóa** | **sp\_XoaLichHen** | **Khóa** |
| ***Input:*** Mã nha sĩ (@MaNhaSi)  ***Output***: Lịch hẹn của nha sĩ @MaNhaSi | ***Input***: Mã nha sĩ (@MaNhaSi), Ngày (@Ngay), Thứ tự ca (@ThuTuCa)  ***Output***: Xóa lịch hẹn của nha sĩ @MaNhaSi, vào ngày @Ngay, thứ tự ca @ThuTuCa |
| SET TRANSACTION ISOLATION  LEVEL **READ COMMITTED** | SET TRANSACTION ISOLATION  LEVEL **READ COMMITTED** |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra thông tin @MaNhaSi có tồn tại không  IF NOT EXISTS (SELECT \* FROM NHASI WHERE NHASI.MaNhaSi=@MaNhaSi)  BEGIN  PRINT (N‘Nha sĩ không tồn tại’)  ROLLBACK TRAN  RETURN 1  END | **R(NHASI)**  //Xin khoá đọc trên bảng NHASI |  |  |
| WAITFOR DELAY '00:00:20' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: Kiểm tra thông tin @MaNhaSi, @Ngay, @ThuTuCa có tồn tại trong bảng LICHCANHAN không  IF NOT EXISTS (SELECT \* FROM LICHCANHAN WHERE LICHCANHAN.MaNhaSi=@MaNhaSi AND LICHCANHAN.Ngay = @Ngay AND LICHCANHAN.ThuTuCa = @ThuTuCa)  BEGIN  PRINT(N’Lịch cá nhân này không tồn tại’)  ROLLBACK TRAN  RETURN 1  END | **R(LICHCANHAN)**  //Xin khóa đọc trên bảng LICHCANHAN |
|  |  | B2: Thực hiện xóa lịch cá nhân cần xóa  DELETE LICHCANHAN ~~WITH (NOLOCK)~~  WHERE MaNhaSi = @MaNhaSi  AND Ngay = @Ngay  AND ThuTuCa = @ThuTuCa | **X(LICHCANHAN)**  //Xin khóa khi trên bảng LICHCANHAN |
|  |  | WAITFOR DELAY '00:00:20' |  |
| B2: Xem lịch cá nhân của nha sĩ @MaNhaSi  SELECT \*  FROM LICHCANHAN  WHERE MaNhaSi = @MaNhaSi | **R(LICHCANHAN)**  //Xin khoá đọc trên bảng LICHCANHAN |  |  |
| COMMIT |  |  |  |
|  |  | COMMIT |  |

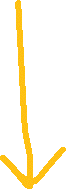
# Sinh viên thực hiện: 21120551 – Đàm Tử Tâm

## Tình huống 1: Dirty Read



|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: *Dirty read***  T1 (User = Nhân viên): thực hiện thêm lịch hẹn  T2 (User = Nha sĩ): thực hiện xem lịch hẹn của mình trong ngày được chọn | | | |
| **sp\_ThemLichHen** | **Khóa** | **sp\_XemLichHen** | **Khóa** |
| ***Input:*** Mã phiếu hẹn (@MaPhieu), mã nha sĩ khám (@NhaSiKham), ngày khám (@Ngay), thứ tự ca khám (@ThuTuCa), số điện thoại khách hàng (@SdtKhachHang)  ***Output***: Lịch hẹn không được thêm vào bảng PHIEUHEN do vi phạm RBTV (không được hẹn lịch cá nhân đang trong trạng thái ‘Bận’), thực hiện rollback transaction | ***Input***: Mã nha sĩ (@MaNhaSi), ngày muốn xem (@Ngay)  ***Output***: Danh sách lịch hẹn của nha sĩ được lọc theo mã nha sĩ và ngày, xử lý được việc lịch hẹn bị trùng do lỗi Dirty Read |
| SET TRANSACTION ISOLATION  LEVEL **READ COMMITTED** | SET TRANSACTION ISOLATION  LEVEL **READ COMMITTED**  ***Giải thích***: Dù giao tác T2 chỉ có lệnh đọc, không có lệnh ghi, không cần phát khóa S để ngăn chặn hành động ghi nên ta đã dùng mức cô lập **READ UNCOMMITTED** đểtăng tốc độ hệ thống. Tuy nhiên, mức cô lập này gây lỗi Dirty Read. Để khắc phục lỗi này, ta sử dụng mức cô lập **READ COMMITTED** nhằm ngăn việc đọc dữ liệu chưa được commit/rollback. |
| BEGIN TRAN |  |  |  |
| B1: Lưu lịch cá nhân của nha sĩ trong ngày muốn hẹn vào bảng @@TempLichCaNhan để lúc sau kiểm tra nha sĩ có trong trạng thái ‘Bận’ hay không.  DECLARE @TempLichCaNhan TABLE (  MaNhaSi VARCHAR(10),  Ngay DATE,  ThuTuCa INT,  TrangThai NVARCHAR(20)  );  INSERT INTO @TempLichCaNhan  SELECT\* FROM LICHCANHAN WHERE MaNhaSi=@NhaSiKham AND Ngay=@Ngay AND THUTUCA=@ThuTuCa | **S(LICHCANHAN)**  //Xin khóa đọc trên bảng LICHCANHAN để thực hiện lệnh ‘*SELECT\* FROM LICHCANHAN…*’ và **xin thành công** do LICHCANHAN không giữ khóa khác . **Khóa này được trả khóa ngay sau khi đọc xong** chứ không giữ đến cuối giao tác (do mức cô lập READ COMMITTED). |  |  |
| B2: Thêm lịch hẹn vào bảng PHIEUHEN  INSERT INTO PHIEUHEN VALUES (@MaPhieu,@NhaSiKham,@Ngay,  @ThuTuCa,@SdtKhachHang) | **X(PHIEUHEN)**  //Xin khoá ghi trên bảng PHIEUHEN để insert vào PHIEUHEN, xin thành công do PHIEUHEN không giữ khóa khác, **khóa này được giữ đến cuối giao tác**. |  |  |
| WAITFOR DELAY '00:00:05' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: Kiểm tra xem có @MaNhaSi tồn tại trong bảng NHASI không. Nếu không thì in ra thông báo không tồn tại nha sĩ và Rollback, ngược lại sang B2.  IF (NOT EXISTS (SELECT\*FROM NHASI WHERE MaNhaSi=@MaNhaSi))  BEGIN  PRINT(N'NHA SĨ KHÔNG TỒN TẠI')  **ROLLBACK TRAN**  RETURN 1  END | **S(NHASI)**  //Xin khoá đọc trên bảng NHASI để thực hiện ‘*SELECT\*FROM NHASI…*’ và xin thành công do NHASI không giữ khóa khác. K**hóa này được trả khóa ngay sau khi đọc xong** do mức cô lập là READ COMMITTED |
|  |  | B2: Kiểm tra xem Nha sĩ @MaNhaSi có lịch hẹn trong ngày @Ngay không. Nếu không thì in ra thông báo Nha sĩ không có lịch hẹn trong ngày rồi Rollback, ngược lại sang B3.  IF (NOT EXISTS (SELECT\*FROM PHIEUHENWHERE NhaSiKham=@MaNhaSi AND Ngay=@Ngay))  BEGIN  PRINT(N'NHA SĨ KHÔNG CÓ LỊCH HẸN TRONG NGÀY' + cast(@Ngay as VARCHAR))  **ROLLBACK TRAN**  RETURN 1  END | **S(PHIEUHEN)**  //Xin khóa s trên bảng PHIEUHEN để thực hiện ‘*SELECT\*FROM PHIEUHEN…*’, tuy nhiên **xin không thành công** do PHIEUHEN đang giữ khóa X ở T1, do đó phải đợi T1 commit/rollback mới xin được khóa S ở đây, **khóa này được trả khóa ngay sau khi đọc xong** do mức cô lập là READ COMMITTED |
|  |  | B3: Hiển thị danh sách lịch hẹn của Nha sĩ có mã @MaNhaSi trong ngày @Ngay  SELECT \* FROM PHIEUHEN WHERE NhaSiKham=@MaNhaSi AND Ngay=@Ngay | **S(PHIEUHEN)**  //Xin khóa s trên bảng PHIEUHEN để thực hiện ‘*SELECT\*FROM PHIEUHEN…*’, tuy nhiên **xin không thành công** do PHIEUHEN đang giữ khóa X ở T1, do đó phải đợi T1 commit/rollback mới xin được khóa S ở đây, **khóa này được trả khóa ngay sau khi đọc xong** do mức cô lập là READ COMMITTED |
| B3: Kiểm tra xem lịch cá nhân của Nha sĩ vừa được thêm vào lịch hẹn có ở trạng thái ‘Bận’ hay không? Nếu có thì rollback transaction  IF EXISTS (SELECT \* FROM @TempLichCaNhan WHERE TrangThai = N'Bận')  BEGIN  PRINT(N'LỊCH CỦA NHA SĨ NÀY ĐÃ BẬN, VUI LÒNG CHỌN LỊCH KHÁC')  ROLLBACK  RETURN 1  END | //Không cần khóa do sử dụng biến table | COMMIT |  |

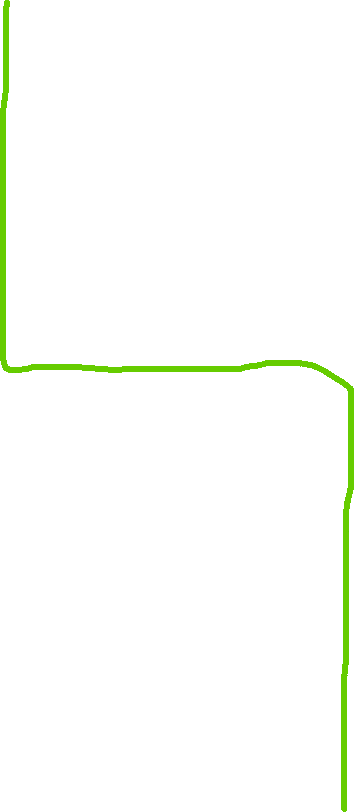
Nét vàng: đường đi mong muốn , Nét xanh: đường đi thật sự , Nét đỏ: thời gian khóa được giữ



Một số lưu ý:

* Để xảy ra và xử lý được Dirty Read, cần demo **@Ngay và @MaNhaSi ở T2** sẽ giống **@Ngay và @NhaSiKham ở T1**
* Theo như em tìm hiểu, các biến table (table variable) không dùng cơ chế phát khóa giống các bảng thông thường trong một cơ sở dữ liệu, người dùng cũng không thể kiểm soát cơ chế phát khóa của biến table thông qua mức cô lập. Do đó, biến table @TempLichCaNhan sẽ không phát khóa X hay khóa S nên trong quá trình thực hiện giao tác sẽ không có trình bày khóa cho biến table này.

## Tình huống 2: Unrepeatable read



|  |  |  |  |
| --- | --- | --- | --- |
| **ERR02: *Unrepeatable read***  T1 (User = Nha sĩ): thực hiện xem lịch hẹn của mình trong ngày được chọn  T2 (User = Nhân viên): thực hiện sửa lịch hẹn | | | |
| **sp\_** **XemLichHen** | **Khóa** | **sp\_SuaLichHen** | **Khóa** |
| ***Input:*** Mã nha sĩ (@MaNhaSi), Ngày (@Ngay). Ngày được chọn của nha nghĩ @MaNhaSi chỉ có 1 lịch hẹn duy nhất trong ngày để thể hiện và xử lý được Unrepeatable Read  ***Output***: Lịch hẹn của nha sĩ trong ngày được chọn trước khi hành động sửa lịch hẹn ở T2 được thực hiện | ***Input***: Mã phiếu (@MaPhieu), mã nha sĩ (@NhaSiKham), ngày khám (@Ngay), ca khám (@ThuTuCa), số điện thoại khách hàng (@SdtKhachHang)  ***Output***: sửa lịch hẹn thành công, lịch hẹn của Nha sĩ trong ngày cũ sẽ trống do đã được dời lịch sang ngày mới hoặc đổi sang Nha sĩ khác |
| SET TRANSACTION ISOLATION  LEVEL **REPEATABLE READ** | SET TRANSACTION ISOLATION  LEVEL **READ COMMITTED** |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra xem có @MaNhaSi tồn tại trong bảng NHASI không. Nếu không thì in ra thông báo không tồn tại nha sĩ và Rollback, ngược lại sang B2.  IF (NOT EXISTS (SELECT\*FROM NHASI WHERE MaNhaSi=@MaNhaSi))  BEGIN  PRINT(N'NHA SĨ KHÔNG TỒN TẠI')  **ROLLBACK TRAN**  RETURN 1  END | **S(NHASI)**  //Xin khoá đọc trên bảng NHASI để thực hiện ‘*SELECT\*FROM NHASI…*’ và xin thành công do NHASI không giữ khóa khác. K**hóa này được giữ đến hết giao tác** do mức cô lập là REPEATABLE READ. |  |  |
| B2: Kiểm tra xem Nha sĩ @MaNhaSi có lịch hẹn trong ngày @Ngay không. Nếu không thì in ra thông báo Nha sĩ không có lịch hẹn trong ngày rồi Rollback, ngược lại tiếp tục transaction.  IF (NOT EXISTS (SELECT\*FROM PHIEUHENWHERE NhaSiKham=@MaNhaSi AND Ngay=@Ngay))  BEGIN  PRINT(N'NHA SĨ KHÔNG CÓ LỊCH HẸN TRONG NGÀY' + cast(@Ngay as VARCHAR))  **ROLLBACK TRAN**  RETURN 1  END | **S(PHIEUHEN)**  //Xin khoá đọc trên bảng PHIEUHEN để thực hiện ‘*SELECT\*FROM PHIEUHEN…*’, xin thành công do PHIEUHEN không giữ khóa khác. **Khóa này được giữ đến hết giao tác** do mức cô lập là REPEATABLE READ. |  |  |
| WAITFOR DELAY '00:00:05' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: Cập nhật thông tin lịch hẹn, chuyển lịch hẹn hiện tại sang một ngày khác hoặc một Nha sĩ khác hợp lệ  UPDATE PHIEUHEN  SET NhaSiKham=@NhaSiKham, Ngay=@Ngay,  ThuTuCa=@ThuTuCa,SdtKhachHang=@SdtKhachHang  WHERE MaPhieu=@MaPhieu | **X(PHIEUHEN)**  //Xin khóa X trên bảng PHIEUHEN để thực hiện lệnh UPDATE trên bảng PHIEUHEN nhưng **xin không thành công** do hiện tại bảng PHIEUHEN đang giữ khóa S ở T1 mà khóa X và S không tương thích. Do đó, phải đợi T1 commit/rollback mới xin được khóa S ở đây, **khóa này được giữ đến hết giao tác.** |
| B3: Hiển thị danh sách lịch hẹn của Nha sĩ có mã @MaNhaSi trong ngày @Ngay  SELECT \*  FROM PHIEUHEN  WHERE NhaSiKham=@MaNhaSi AND Ngay=@Ngay | **S(PHIEUHEN)**  //Xin khoá đọc trên bảng PHIEUHEN để thực hiện ‘*SELECT\*FROM PHIEUHEN…*’, xin thành công do PHIEUHEN chỉ đang giữ khóa S ở T1 và hai khóa S tương thích với nhau. **Khóa này được giữ đến hết giao tác** do mức cô lập là REPEATABLE READ. |  |  |
| COMMIT |  | COMMIT |  |

Nét vàng: đường đi mong muốn , Nét xanh: đường đi thật sự , Nét đỏ: thời gian khóa được giữ

Một số lưu ý:

* Để thể hiện và xử lý được Unrepeatable read, số lượng lịch hẹn của Nha sĩ trong ngày muốn cập nhật chỉ là 1. Sau cập nhật, lịch hẹn trong ngày đó của nha sĩ sẽ hoàn toàn trống (do đã chuyển lịch hẹn cũ sang ngày khác hoặc chuyển sang một nha sĩ khác)

# Sinh viên thực hiện: 21120555 – Nguyễn Hữu Thắng

## Tình huống 1: Dirty read

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ERR01: *Dirty read***  T1 (User = Quản trị viên): thực hiên thêm thông tin chi tiết của thuốc  T2 (User = Quản trị viên): thực hiện xem thông tin chi tiết của thuốc | | | | | |
| **sp\_ThemThongTinThuoc** | **Khóa** | **sp\_XemThongTinThuoc** | **Khóa** | | |
| ***Input:*** Mã thuốc (@MaThuoc), Tên thuốc (@TenThuoc), Đơn vị tính (@DonViTinh), Chỉ định (@ChiDinh), Số lượng tồn (@SoLuongTon), Ngày hết hạn (@NgayHetHan)  ***Output***: Thông tin thuốc không được them vào bảng THUOC do vi phạm RBTV |  | | | ***Input***: Mã thuốc @MaThuoc  ***Output***: Chi tiết của một thuốc bao gồm các thuộc tính tương ứng, nếu không tồn tại thì giao tác không được thực hiện |  |
| SET TRANSACTION ISOLATION  LEVEL **READ COMMITED** |  | | | SET TRANSACTION ISOLATION  LEVEL **READ COMMITED** |  |
| BEGIN TRAN |  |  |  | | |
| B1: Thêm thông tin thuốc vào bảng THUOC    INSERT INTO THUOC VALUES (@MaThuoc, @TenThuoc, @DonViTinh, @ChiDinh, @SoLuongTon, @NgayHetHan) | **X(THUOC)**  //Xin khoá ghi trên bảng THUOC để insert vào THUOC, xin thành công do THUOC không giữ khóa khác, khóa này được giữ đến cuối giao tác |  |  | | |
| WAITFOR DELAY '00:00:20' |  |  |  | | |
|  |  | BEGIN TRAN |  | | |
|  |  | B1: Kiểm tra thông tin mã thuốc có tồn tại  IF NOT EXISTS (SELECT \* FROM THUOC WHERE MaThuoc = @MaThuoc)  BEGIN  PRINT (N ‘Mã thuốc không tồn tại’)  ROLLBACK TRAN  RETURN 1  END | **R(THUOC)**  //Xin khóa đọc trên bảng thuốc, do mức cô lập là read commited, không được cấp khóa vì phải chờ giao tác giữ khóa phải trả khóa ghi đang còn trên bảng THUOC | | |
|  |  | B2: Xuất các thông tin của thuốc đó  SELECT \* FROM THUOC  WHERE MaThuoc = @MaThuoc | **R(THUOC)**  //Xin khóa đọc trên bảng thuốc, do mức cô lập là read commited, không được cấp khóa vì phải chờ giao tác giữ khóa phải trả khóa ghi đang còn trên bảng THUOC | | |
|  |  | COMMIT |  | | |
| B2: Kiểm tra ngày hết hạn có hợp lệ  IF DATEDIFF(DD,@NgayHetHan,GETDATE()) > 0  BEGIN  PRINT N'Ngày hết hạn phải sau ngày hiện tại'  ROLLBACK TRAN  RETURN 1  END |  |  |  | | |
|  | | |  |  |  |

Link các đường đi: https://drive.google.com/file/d/1rOWUAUyvEyH9zT86FtTpMU4l4UyMwky-/view?usp=sharing

Nét vàng: đường đi mong muốn , Nét xanh: đường đi thật sự , Nét đỏ: thời gian khóa được giữ

## Tình huống 2: Unrepeatble read

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ERR01: Unrepeatable read**  T1 (User = Nhân viên): thực hiện thêm một phiếu hẹn khám (đăng kí hẹn khám)  T2 (User = Nha Sĩ): thực hiện xóa một lịch cá nhân – chưa được hẹn | | | | | | |
| **sp\_ThemPhieuHen** | **Khóa** | **sp\_XoaLichCaNhan** | **Khóa** | | | |
| ***Input:*** Mã Phiếu(@Ma ), Mã nha sĩ (@NhaSi), Ngày(@Ngay), thứ tự ca (@ca), số điện thoại khách hàng (@sdt).  ***Output***: Thông báo thêm lịch hẹn thành công/thất bại |  | | ***Input***: Mã nha sĩ (@MaNhaSi), Ngày (@ngay), Ca trực (@ca)  ***Output***: Xóa lịch cá nhân thành công | |  | |
| SET TRANSACTION ISOLATION  LEVEl **REPEATABLE READ** |  | | SET TRANSACTION ISOLATION  LEVEL **READ COMMITTED** | |  | |
| BEGIN TRAN |  |  |  | | | |
| B1: Kiểm tra xem Mã phiếu đã tồn tại chưa.  IF EXISTS (SELECT \* FROM PHIEUHEN WHERE MaPhieu = @Ma)  BEGIN  PRINT N'Đã tồn tại phiếu hẹn có mã: ' + @Ma  ROLLBACK TRAN  RETURN 1  END | **S(PHIEUHEN)**Xin khoá đọc trên Bảng PhieuHen, do trước đó chưa có khóa nào đặt lên bảng PhieuHen nên được phát ngay và giữ khóa cho đến hết giao tác do mức cô lập là Repeateable read. |  |  | | | |
| B2: Kiểm tra xem có lịch cá nhận nào phù hợp để sắp xếp hẹn theo thông tin input hay không  IF NOT EXISTS(SELECT \*  FROM LICHCANHAN  WHERE Ngay = @Ngay AND ThuTuCa = @Ca AND MaNhaSi = @Nhasi AND TrangThai <> N'Bận')  BEGIN  PRINT N'Lịch hẹn không hợp lệ với lịch cá nhân của nha sĩ'  ROLLBACK TRAN  RETURN 1  END | **S(LICHCANHAN)**  Xin khoá đọc trên bảng LICHCANHAN, do trước đó chưa có khóa nào đặt lên đơn vị dữ liệu này nên được phát ngay và giữ khóa cho đến hết giao tác do mức cô lập là Repeateable read. |  |  | | | |
| B3: Kiểm tra xem Lịch cá nhân được chọn đã được đặt khám chưa.  IF EXISTS (SELECT \*  FROM PHIEUHEN WHERE Ngay = @Ngay and ThuTuCa = @Ca and NhaSiKham = @Nhasi)  BEGIN  PRINT N'Buổi làm việc này đã được một khách hàng khác đăng kí hẹn.'  ROLLBACK TRAN  RETURN 1  END | **S(PHIEUHEN)** Xin khóa đọc trên đơn vị dữ liệu PHIEUHEN, trước đó PHIEUHEN giữ khóa đọc và trả khóa ngay sau đó nên hiện tại không có khóa nào đặt lên đơn vị dữ liệu này. Giữ khóa cho đến hết giao tác do mức cô lập là Repeateable read.. |  |  | | | |
| WAITFOR DELAY '00:00:05' |  |  |  | | | |
|  |  | BEGIN TRAN |  | | | |
|  |  | B1: Kiểm tra thông tin có tồn tại Nha sĩ có mã nha sĩ input hay không.  IF NOT EXISTS(SELECT \*  FROM NHASI  WHERE MaNhaSi = @MaNhaSi)  BEGIN  PRINT N'Không tồn tại nha sĩ'  ROLLBACK TRAN  RETURN 1  END | **S(NHASI)**  //Xin khóa đọc trên đơn vị dữ liệu NHASI, được cấp khóa và trả khóa ngay sau khi đọc (do NHASI hiện không giữ khóa và mức cô lập là Read committed) | | | |
|  |  | B2: Kiểm tra thông tin LICHTRUC có hợp lệ hay không.  IF NOT EXISTS(SELECT \*  FROM LICHTRUC  WHERE Ngay = @Ngay and ThuTuCa = @Ca)  BEGIN  PRINT N'Lịch trực không tồn tại'  ROLLBACK TRAN  RETURN 1  END | **S(LICHTRUC)**  Xin khóa đọc trên bảng LICHTRUC, được cấp khóa ngay và trả khóa ngay sau khi đọc xong. (do LICHTRUC hiện không giữ khóa và mức cô lập là Read committed) | | | |
|  |  | B2: Xóa lịch cá nhân  DELETE LICHCANHAN  WHERE MaNhaSi = @MaNhaSi AND Ngay = @Ngay AND ThuTuCa = @Ca | **X(LICHCANHAN)**  Xin khóa ghi trên đơn vị dữ liệu LICHCANHAN, được cấp khóa và giữ khóa đến khi giao tác kết thúc (rollback hoặc committ) | | | |
| INSERT INTO PHIEUHEN  VALUES (@Ma, @Nhasi, @Ngay, @Ca, @sdt) | **X(PHIEUHEN)**  //Xin khóa ghi để thực hiện thêm một PHIEUHEN, được cấp khóa ghi trên đơn vị dữ liệu PHIEUHEN và giữ đến hết giao tác. |  |  | | | |
| UPDATE LICHCANHAN  SET TrangThai = N'Bận'  WHERE Ngay = @Ngay and ThuTuCa = @Ca and MaNhaSi = @Nhasi | **X(LICHCANHAN)**  //Xin khóa ghi để thực hiện UPDATE trạng thái, chờ để được cấp khóa do LICHCANHAN đang giữ khóa ghi của giao tác khác. |  |  | | | |
|  | | |  | COMMIT | |  |
| COMMIT | | |  |  | |  |
|  | | |  |  | |  |

Link các đường đi: https://drive.google.com/file/d/1wz-fnC6OmbEhHOgR85cnConprwtwecIE/view?usp=sharing

Nét vàng: đường đi mong muốn , Nét xanh: đường đi thật sự , Nét đỏ: thời gian khóa được giữ

# Sinh viên thực hiện: 21120582 – Đinh Hoàng Trung

## Tình huống 1: Lost update



|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: Lost update**  T1 (User = Nha Sĩ): thực hiện thêm 1 thông tin một chi tiết đơn thuốc cho hồ sơ bệnh nhân  T2 (User = Quản trị viên): thực hiện cập nhật số lượng tồn kho của thuốc. | | | |
|  | **Khóa** |  | **Khóa** |
| ***Input:*** Mã Thuốc (@MaThuoc), số điện thoại khách hàng (@sdt), Mã bệnh án (@MaBA) mã bệnh án phân biệt các bệnh án của 1 khách hàng, Số lượng kê đơn (@soluong)  ***Output***: Thông báo thêm chi tiết đơn thuốc thành công, in ra đơn thuốc, bên cạnh đó thì số lượng thuốc tồn kho sẽ được cập nhật lại ngay sau khi chi tiết đơn thuốc được thêm thành công. Ngược lại, thông báo thất bại nếu thêm khi thành công. | ***Input***: Mã thuốc (@MaThuoc) – Mã thuốc phân biệt giữa các thuốc với nhau, Số lượng cập nhật (@soluogncapnhat) – Số lượng muốn thêm vào hoặc giảm bớt cho số lượng tồn kho.  ***Output***: Thông báo cập nhật thành công, in ra bảng thuốc gồm mã, tên và số lượng tồn kho của các loại thuốc. |
| SET TRANSACTION ISOLATION  LEVEL **READ COMMITTED** | SET TRANSACTION ISOLATION  LEVEL **READ COMMITTED** |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra thông tin (1) ChiTietDonThuoc đã được kê vào bệnh án hay chưa.  if exists(select \*  from CHITIETDONTHUOC  where MaThuoc = @MaThuoc and MaBenhAn = @MaBA and SdtKhachHang = @sdt)  begin  print N'Chi tiết đơn thuốc đã tồn tại - Thuốc đã được kê trong bệnh án ' + @MaBA + N' của khách hàng có số điện thoại ' + @sdt  rollback tran  return 1  end | **S(CHITIETDONTHUOC)**  //Xin khoá đọc trên bảng CHITIETDONTHUOC, vì hiện đơn vị dữ liệu chưa có bất kì khóa nào đặt lên => được cấp khóa, và trả khóa ngay sau đọc xong do mức cô lập là Read committed |  |  |
| B2: Kiểm tra thông tin (2) KhachHang có tồn tại hay không.  if not exists (select \* from KHACHHANG where Sdt = @sdt)  begin  print N'Không tồn tại khách hàng có số điện thoại: ' + @sdt  rollback tran  return 1  end | **S(KHACHHANG)**  //Xin khoá đọc trên bảng KHACHHANG, do đơn vị dữ liệu khách hàng chưa có khóa nào đặt lên nên được cấp khóa và trả khóa ngay sau khi đọc xong. |  |  |
| B3: Kiểm tra thông tin (3) Kiểm tra xem còn đủ thuốc để kê đơn hay không.  if @soluong > (select soluongtonkho from THUOC where MaThuoc = @MaThuoc)  begin  print N'Không còn đủ thuốc ' + @Mathuoc + N' để thực hiện kê đơn.'  rollback tran  return 1  end | **S(THUOC)**  Xin khóa đọc trên bảng THUOC, được cấp kháo ngay do chưa có khóa, trả khóa ngay sau khi hoàn thành việc đọc. |  |  |
| declare @soluongthuoc int =  (select SoLuongTonKho  from THUOC with (rowlock)  where MaThuoc = @MaThuoc)  set @soluongthuoc = @soluongthuoc - @soluong | **S(THUOC)**  Xin khóa đọc trên bảng THUOC, được cấp kháo ngay do chưa có khóa, giữ khóa cho đến hết giao tác và chỉ khóa trên các dòng được select, giao tác khác xin khóa đọc trên các dòng này sẽ phải chờ cho đến hết giao tác mới được cấp. |  |  |
| WAITFOR DELAY '00:00:20' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: Kiểm tra thông tin (1) THUOC có tồn tại thuốc nào có mã thuốc được truyền vào hay không.  if not exists (select \* from THUOC where MaThuoc = @MaThuoc)  begin  print cast(N'Không tồn tại thuốc có mã: ' + @MaThuoc as nvarchar(10))  rollback tran  return 1  end | **S(THUOC)**  //Xin khóa đọc bảng THUOC, do trước đó ở T1 có xin khóa đọc và chỉ khóa các dòng xác định.  => được cấp khóa (trong trường hợp không trùng với các dòng dữ liệu bị khóa) và trả khóa lại ngay sau khi đọc do mức cô lập là Read Committed.  Giả định 2 giao tác đang giao tác trên cùng dòng dữ liệu => phải chờ để được cấp khóa đọc. |
|  |  | B2: Kiểm tra soluongcapnhat – số lượng cập nhật không được bằng 0.  if @soluongcapnhat = 0  begin  print N'Số lượng cập nhật phải khác 0'  rollback tran  return 1  end |  |
|  |  | B3: Kiểm tra số lượng tồn kho sau khi cập nhật có âm hay không.  if (select SoLuongTonKho from THUOC where MaThuoc = @MaThuoc)-@soluongcapnhat < 0  begin  print N'Số lượng tồn kho sau khi cập nhật số lượng không được âm'  rollback tran return 1  end | **S(THUOC)**  //Xin khóa đọc bảng THUOC, do trước đó ở T1 có xin khóa đọc và chỉ khóa các dòng xác định.  => được cấp khóa (trong trường hợp không trùng với các dòng dữ liệu bị khóa) và trả khóa lại ngay sau khi đọc do mức cô lập là Read Committed.  Giả định 2 giao tác đang giao tác trên cùng dòng dữ liệu => phải chờ để được cấp khóa đọc. |
|  |  | declare @soluongthuoc int =  (select SoLuongTonKho  from THUOC with rowlock  where MaThuoc = @MaThuoc)  set @soluongthuoc = @soluongthuoc + @soluongcapnhat | **S(THUOC)**  //Xin khóa đọc bảng THUOC, giả định giao tác này xin khóa đọc trên các dòng dữ liệu trùng với các dòng bị khóa ở giao tác trước.   * Phải chờ cho đến khi giao tác trước hoàn thành (commit hoặc rollback). |
| update THUOC  set SoLuongTonKho = @soluongthuoc  where MaThuoc = @MaThuoc | X(THUOC)  Xin khóa X trên bảng THUOC để thực hiện Update số lượng tồn kho, nâng cấp khóa đọc thành khóa ghi. |  |  |
| insert into CHITIETDONTHUOC  values (@MaThuoc, @sdt, @MaBA, @soluong) | X(CHITIEDONTHUOC)  Xin khóa X trên đơn vị dữ liệu CHITIETDONTHUOC để thực hiện Insert 1 dòng dữ liệu vào bảng, do thời điểm này đơn vị dữ liệu không giữ bất cứ khóa nào nên được cấp khóa và giữ khóa đến cuối giao tác. |  |  |
|  |  | Update THUOC  set SoLuongTonKho = @soluongthuoc  where MaThuoc = @MaThuoc | X(THUOC)  Xin khóa X trên đơn vị dữ liệu THUOC để thực hiện Update SoLuongTonKho, phải phải chờ đến cuối giao tác 1 để được cấp khóa. |
| COMMIT |  |  |  |
|  |  | COMMIT |  |

Kí hiệu đường đi:



* + - * \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_: Thời gian khóa được giữ.
      * \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_: đường đi thật sự.
      * \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_: đường đi mong muốn.

## Tình huống 2: Phantom

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: Phantom**  T1 (User = Nhân viên): thực hiện thêm một phiếu hẹn khám (đăng kí hẹn khám)  T2 (User = Nha Sĩ): thực hiện xóa một lịch cá nhân – chưa được hẹn | | | |
| **sp\_ThemPhieuHen** | **Khóa** | **sp\_Insert\_LichCaNhan** | **Khóa** |
| ***Input:*** Mã Phiếu hẹn(@MaPhieu ), Mã nha sĩ (@NhaSiKham) – nha sĩ phụ trách khách cho buổi hẹn đó, Ngày hẹn (@Ngay), thứ tự ca (@ca) – ca làm việc trong ngày, số điện thoại khách hẹn (@sdt).  ***Output***: Thông báo thêm lịch hẹn thành công/thất bại, kèm theo danh sách lịch hẹn của khách hàng. | ***Input***: Mã nha sĩ (@MaNhaSi), Ngày (@ngay), Ca trực (@ca)  ***Output***: Thông báo thêm lịch cá nhân thành công/thất bại, kèm theo lịch cá nhân của nha sĩ trong tháng tới. |
| SET TRANSACTION ISOLATION  LEVEl **SERIALIZABLE** | SET TRANSACTION ISOLATION  LEVEL **READ COMMITTED** |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra thông tin (1): Kiểm tra xem Mã phiếu đã tồn tại chưa.  if exists (select \* from PHIEUHEN where MaPhieu = @Ma)  begin  print N'Đã tồn tại phiếu hẹn có mã: ' + @Ma  rollback tran  return 1  end | **S(PHIEUHEN)**  //Xin khoá đọc trên Bảng PhieuHen, do trước đó chưa có khóa nào đặt lên bảng PhieuHen nên được phát ngay và giữ khóa cho đến hết giao tác do mức cô lập là Repeateable read. |  |  |
| B2: Kiểm tra thông tin (2): Kiểm tra xem có lịch cá nhận nào phù hợp để sắp xếp hẹn theo thông tin input hay không  if not exists(select \*  from LICHCANHAN  where Ngay = @Ngay and ThuTuCa = @Ca and MaNhaSi = @Nhasi and TrangThai <> N'Bận')  begin  Print N'Lịch hẹn không hợp lệ với lịch cá nhân của nha sĩ'  rollback tran  return 1  end | **S(LICHCANHAN)**  //Xin khoá đọc trên bảng LICHCANHAN, do trước đó chưa có khóa nào đặt lên đơn vị dữ liệu này nên được phát ngay và giữ khóa cho đến hết giao tác do mức cô lập là Repeateable read. |  |  |
| B3: Kiểm tra thông tin (2): Kiểm tra xem Lịch cá nhân được chọn đã được đặt khám chưa.  if exists (select \*  from PHIEUHEN where Ngay = @Ngay and ThuTuCa = @Ca and NhaSiKham = @Nhasi)  begin  print N'Buổi làm việc này đã được một khách hàng khác đăng kí hẹn.'  rollback tran  return 1  end | **S(PHIEUHEN)**  //Xin khóa đọc trên đơn vị dữ liệu PHIEUHEN, trước đó PHIEUHEN giữ khóa đọc và trả khóa ngay sau đó nên hiện tại không có khóa nào đặt lên đơn vị dữ liệu này. Giữ khóa cho đến hết giao tác do mức cô lập là Repeateable read. |  |  |
| WAITFOR DELAY '00:00:05' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: Kiểm tra thông tin (1) LICHCANHAN có bị trùng hay không  if exists (select \*  from LICHCANHAN  where MaNhaSi = @MaNhaSi and Ngay = @Ngay and ThuTuCa = @ca)  begin  print N'Lịch cá nhân đã tồn tại.'  rollback tran  return 1  end | **S(LICHCANHAN)**  //Xin khóa đọc bảng LICHCANHAN, được cấp khóa đọc do giao tác khác đang giữ khóa đọc nhưng 2 khóa đọc không gây xung đột nên được cấp khóa và trả khóa ngay sau khi đọc xong do mức cô lập là readcommitted |
|  |  | B2: Kiểm tra thông tin có tồn tại Nha sĩ có mã nha sĩ input hay không.  if not exists ( select \*  from NHASI  where MaNhaSi = @MaNhaSi)  begin  print N'Không tồn tại nha sĩ có mã: ' + @manhasi  rollback tran  return 1  end | **S(NHASI)**  //Xin khóa đọc trên đơn vị dữ liệu NHASI, được cấp khóa và trả khóa ngay sau khi đọc (do NHASI hiện không giữ khóa và mức cô lập là Read committed) |
|  |  | B3: Kiểm tra thông tin LICHTRUC có hợp lệ hay không.  if not exists ( select \*  from LICHTRUC  where Ngay = @Ngay and ThuTuCa = @Ca)  begin  print N'Lịch trực không tồn tại'  rollback tran  return 1  end | **S(LICHTRUC)**  Xin khóa đọc trên bảng LICHTRUC, được cấp khóa ngay do giao tác khác đang giữ khóa đọc và 2 khóa đọc không gây xung đột nên được cấp khóa và trả khóa ngay sau khi đọc xong do mức cô lập là readcommitted. |
|  |  | insert into LICHCANHAN  values( @MaNhaSi, @Ngay, @Ca, N'Rảnh') | **X(LICHCANHAN)**  Xin khóa ghi trên đơn vị dữ liệu LICHCANHAN, phải chờ do khóa đọc trước đó được đặt lên đơn vị dữ liệuvà giữu cho đến hết giao tác (commit hoặc rollback) nên phải chờ cho giao tác trước hoàn thành thì mới được cấp khóa ghi.  Khi được cấp khóa sẽ giữ khóa đến khi hết giao tác |
| insert into PHIEUHEN  values ( @Ma, @Nhasi, @Ngay, @Ca, @sdt) | X(PHIEUHEN)  //Xin khóa ghi để thực hiện thêm một PHIEUHEN, update khóa đọc lên thành khóa ghi do giao tác đang giữ khóa đọc trên đơn vị dữ liệu PHIEUHEN. |  |  |
| update LICHCANHAN  set TrangThai = N'Bận'  where Ngay = @Ngay and ThuTuCa = @Ca and MaNhaSi = @Nhasi | X(LICHCANHAN)  //Xin khóa ghi để thực hiện UPDATE trạng thái, update khóa đọc lên thành khóa ghi do giao tác này đang giữ khóa đọc của LICHCANHAN. |  |  |
|  |  | COMMIT |  |
| COMMIT |  |  |  |

Kí hiệu nét vẽ:

* + - * \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_: Thời gian khóa được giữ.
      * \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_: đường đi thật sự.
      * \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_: đường đi mong muốn.