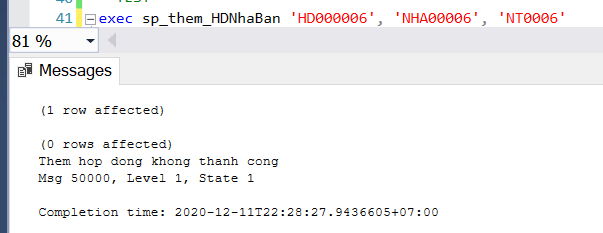
|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: *Dirty read***  T1 (User = NhanVien): Thực hiện thêm một hợp đồng (NV có mã là ‘NV000006’)  T2 (User = NguoiThue): Thực hiện thống kê NhaBan  VD: @maHD=’HD000006’, @maNha=’NHA00006’, @maNT=’NT06’  Sửa lại mức cô lập để không bị lỗi Dirty Read -> T2 sẽ có mức cô lập mời là READ COMMITED để không đọc dữ liệu rác khi T1 bị rollback | | | |
| sp\_them\_HDNhaBan | **Khóa** | sp\_xem\_NhaBan | **Khóa** |
| ***Input:*** @m*aHD, @maNha và @maNT*  ***Output***: Cập nhật lại tình trạng của căn nhà được bán và thêm 1 hợp đồng mới | ***Input***:  ***Output***: Danh sách NhaBan |
| SET TRANSACTION ISOLATION  LEVEL **READ COMMITED** | SET TRANSACTION ISOLATION  LEVEL **READ COMMITED** |
| BEGIN TRAN |  |  |  |
| B1: Cập nhật tình trạng căn nhà được bán  update NhaBan set TinhTrangBan = 0 where MaNha=@maNha | **X(NhaBan)**  Xin khoá ghi trên bảng NhaBan |  |  |
| B2: Thêm 1 dòng mới vào bảng Hợp Đồng  insert into HopDong(MaHD,LoaiHD,ThoiGian,MaNT,MaNha) values (@maHD,0,GETDATE(),@maNT,@maNha) | **X(HopDong)**  Xin khoá ghi trên bảng HopDong |  |  |
| WAITFOR DELAY '00:00: 07' |  |  |  |
| if not exists(select \* from NhaBan where MaNha=@maNha) or not exists(select \* from NguoiThue where MaNT=@maNT)  begin    RAISERROR('Them hop dong khong thanh cong',1,1)  ROLLBACK TRAN sp\_them\_HDNhaBan  end  else  COMMIT TRAN sp\_them\_HDNhaBan |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: Thống kê Nhân viên  select \* from NhaBan | **R(NhaBan)**  Xin khóa đọc trên bảng NhaBan giữ đến hết lệnh |
|  |  | COMMIT |  |

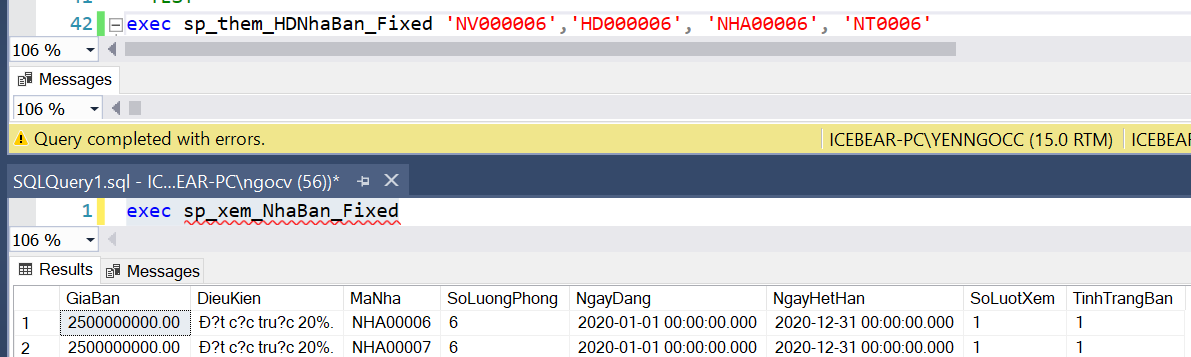
Kết quả:

Hợp đồng không được thêm thành công tức ‘NHA00006’ chưa được bán (TinhTrangNha=1) 🡪 T2 đợi T1 xxong thì mới đọc dữ liệu thì tình trạng nhà của ‘NHA00006’ 1 nghĩa là chưa bán 🡪 Đã sửa Dirty Read

T1:

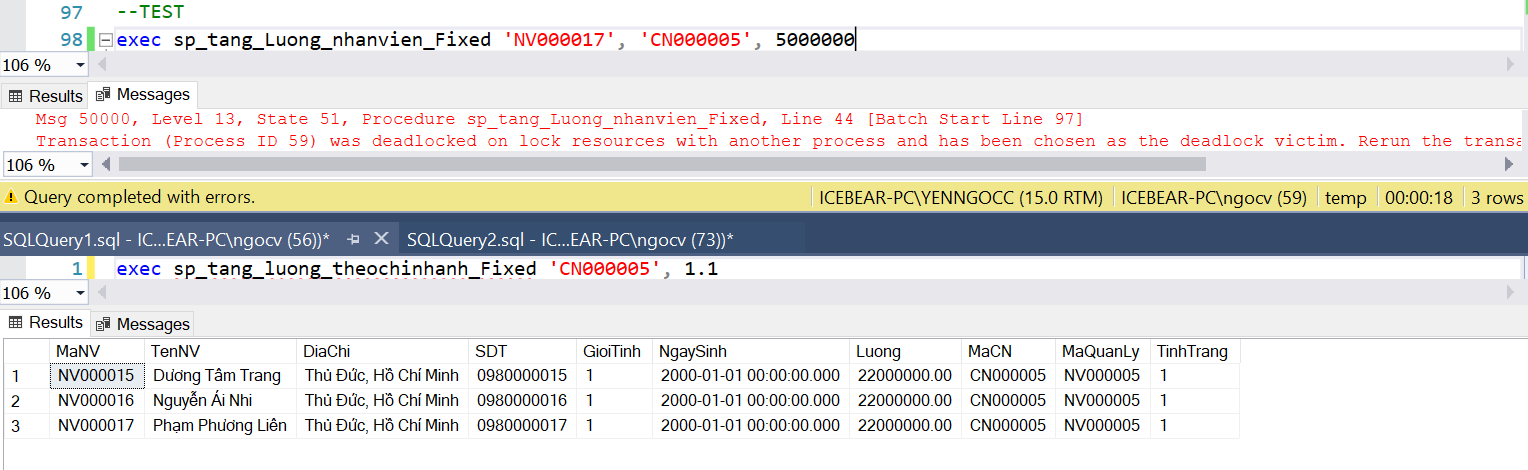


T2:



|  |  |  |  |
| --- | --- | --- | --- |
| **ERR02: *Lost Update***  T1 (User = NhanVienQuanLy): Thực hiện tăng lương của 1 nhân viên trong chi nhánh lên 5tr  T2 (User = CEO): Thực hiện tăng hệ số lương của 1 chi nhánh  VD:  T1: NhanVienQuanLy chi nhánh có maCN=’CN000005’ thay đổi lương của nhân viên có maNV=’NV000017’ tăng lên 1 khoảng @Luong=5tr  T2: CEO thay đổi lương của toàn bộ chi nhánh có maCN=’CN000005’ lên 1.1 lần  Ở VD ta sẽ sửa lại mức cô lập ở T1 là UNREPEATABLE để việc đọc dữ liệu lên của giao tác T1 sẽ giữ khóa đọc suốt cả transaction 🡪 tránh việc đọc dữ liệu lên từ T2.  Sau khi sửa lỗi thì ta gặp một lỗi khác đó là DEADLOCK cụ thể là Conversion deadlock.  Để khắc phục ta thêm một UPD LOCK vào việc đọc dữ liệu của T1 từ bảng Nhân Viên để giữ khóa đọc và chuẩn bị được update lên khóa ghi 🡪 Để hệ thống không cung cấp khóa ghi cho T2 gây ra Deadlock. | | | |
| sp\_tang\_Luong\_nhanvien | **Khóa** | sp\_tang\_luong\_theochinhanh | **Khóa** |
| ***Input:*** @m*aNV, @maCN và @Luong*  ***Output***: Cập nhật lại Luong của nhân viên đó là Luong= Luong+@Luong | ***Input***: @maCN, @heso  ***Output***: Luong của cả chi nhánh có mã là @maCN sẽ tăng lên @heso lần |
| SET TRANSACTION ISOLATION  LEVEL **REPEATABLE READ** | SET TRANSACTION ISOLATION  LEVEL **REPEATABLE READ** |
| BEGIN TRAN |  |  |  |
| B1: Đọc Luong của Nhân viên cần tăng  declare @setluong money  select @setluong=Luong from NhanVien with (UPDLOCK) where MaNV=@maNV | **R(NhanVien)**  Lấy khóa đọc trên bảng NhanVien  đến hết T1 và báo với hệ thống rằng sẽ ghi dữ liệu |  |  |
| WAITFOR DELAY '00:00: 07' |  |  |  |
| B2: Tăng lương của Nhân viên thuộc Chi nhánh lên một lượng là @Luong  set @setluong=@setluong+@Luong  update NhanVien set Luong = @setluong where MaNV=@maNV | **X(NhanVien)**  Khóa ghi ở bảng NhanVien |  |  |
| COMMIT |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B2: Tăng lương của Nhân viên thuộc Chi nhánh theo hệ số  update NhanVien set Luong = Luong\*@hesoLuong where MaCN=@maCN | **X(NhanVien)**  Khóa ghi ở bảng NhanVien |
|  |  | COMMIT |  |

Kết quả sau khi sửa mức cô lập bị Conversion Deadlock làm cho RollBack T1 chỉ thực hiện T2 🡪 do MS SQL Sever tự xử lý Deadlock:



Sau khi thêm UPDLOCK 🡪 Đã sửa và dữ liệu không bị mất việc update của cả 2 Transaction

