BÁO CÁO ĐỒ ÁN MÔN HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU

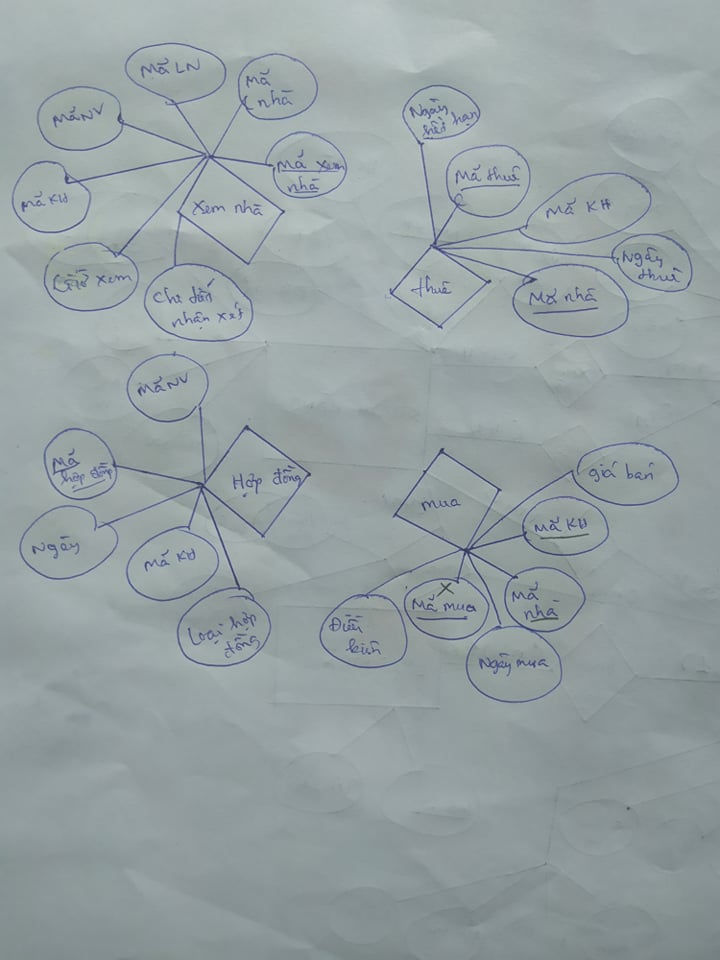
ĐỀ TÀI: QUẢN LÝ HỆ THỐNG CHO THUÊ / BÁN NHÀ

* Thông tin thành viên nhóm:
* Tên nhóm: 18VP-03
* Thành viên:
* **18126037 – ĐINH THÀNH VIỆT**
* **18126029 – NGUYỄN SANG**

Bảng thiết kế ER:

**Ảnh có chứa văn bản, bảng trắng

Mô tả được tạo tự động**

****

1. Lượt đồ quan hệ

* Chi nhánh

|  |  |  |  |
| --- | --- | --- | --- |
| MÃCN | SĐT | ĐỊACHỈ | SỐFAX |

* Nhân viên

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| MÃNV | SĐT | HỌTÊN | GIỚITÍNH | NGÀYSINH | LƯƠNG | MÃCN |

* Khách hàng

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| MÃKH | HỌTÊN | SĐT | ĐỊACHỈ | GIỚITÍNH | THÔNGTINYÊUCẦU | TIÊUCHÍ | MÃCN | MÃNV |

* Loại nhà

|  |  |
| --- | --- |
| MÃLN | TÊNLN |

* Chủ nhà

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MÃCHỦNHÀ | HỌTÊN | LOẠI | SĐT | ĐỊACHỈ |

* Nhà

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| MÃNHÀ | ĐỊACHỈ | TIỀNTHUÊ | THÔNGTIN | NGÀYĐK | NGÀYHẾTHẠN | SỐPHÒNG | SỐLẦNXEM | MÃCHỦNHÀ | MÃCN | MÃNV |

* Mua nhà

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| MÃMUA | MÃNHÀ | MÃKH | GIÁBÁN | NGÀYMUA | ĐIỀUKIỆN |

* Lịch sử xem nhà

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| MÃLSXEMNHA | MÃNV | MÃNHÀ | MÃLN | MÃKH | NGÀYXEM | NHẬNXÉT |

* Bài đăng

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MÃBÀIĐĂNG | LOẠI | MÃNHÀ | MÃCHỦNHÀ | NGÀYĐĂNG |

* Hợp đồng

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MÃHD | MÃNV | MÃKH | NGÀYLẬP | LOẠI |

* Thuê nhà

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MÃTHUÊ | MÃNHÀ | MÃKH | NGÀYĐĂNGKÝ | NGÀYHẾTHẠN |

1. Ràng buộc toàn vẹn
   1. **R1: Mỗi nhân viên chỉ có một mã nhân viên, mã nhân viên không được để trống**

∀ nv1, nv2 ∈ NHANVIEN ( nv1 ≠ nv2 ^ nv1.maNV ≠ nv2.maNV )

* 1. **R2: Mỗi nhân viên chỉ làm việc cho một chi nhánh**
  2. **R3: Giới tính của nhân viên chỉ có thể là nam hoặc nữ**

∀ nv ∈ NHANVIEN ( nv.gioitinh ∈ { ‘Nam’, ‘Nữ’ } )

* 1. **R4: Mỗi loại nhà sẽ có nhiều nhà thuộc loại này**

∀ loainha ∈ LOAINHA ( ∃ nha ∈ NHA ( loainha.maLN = nha.maLN ))

* 1. **R5: Mỗi chi nhánh có một mã chi nhánh, mỗi nhà có một mã nhà**
  2. **R6: Mỗi chủ nhà phải có nhà (nhiều nhà) để bán hoặc cho thuê**

∀ chunha ∈ CHUNHA ( ∃ nha ∈ NHA ( chunha.maNha = nha.maNha ))

* 1. **R7: Mỗi nhà thuộc một loại nhà nào đó**
  2. **R8: Mỗi nhà có thể có nhiều khách đến xem**

∀ khach1, khach2 ∈ KHACHHANG ( khach1.maNha ≠ khach2.maNha v khach1.maNha = khach2.maNha ))

* 1. **R9: Mỗi người có thể xem nhiều nhà vào những ngày khác nhau**
  2. **R10: Ngày đăng phải trước ngày hết hạn**

∀ nha ∈ NHA (nha.ngaydang < nha.ngayhethan)

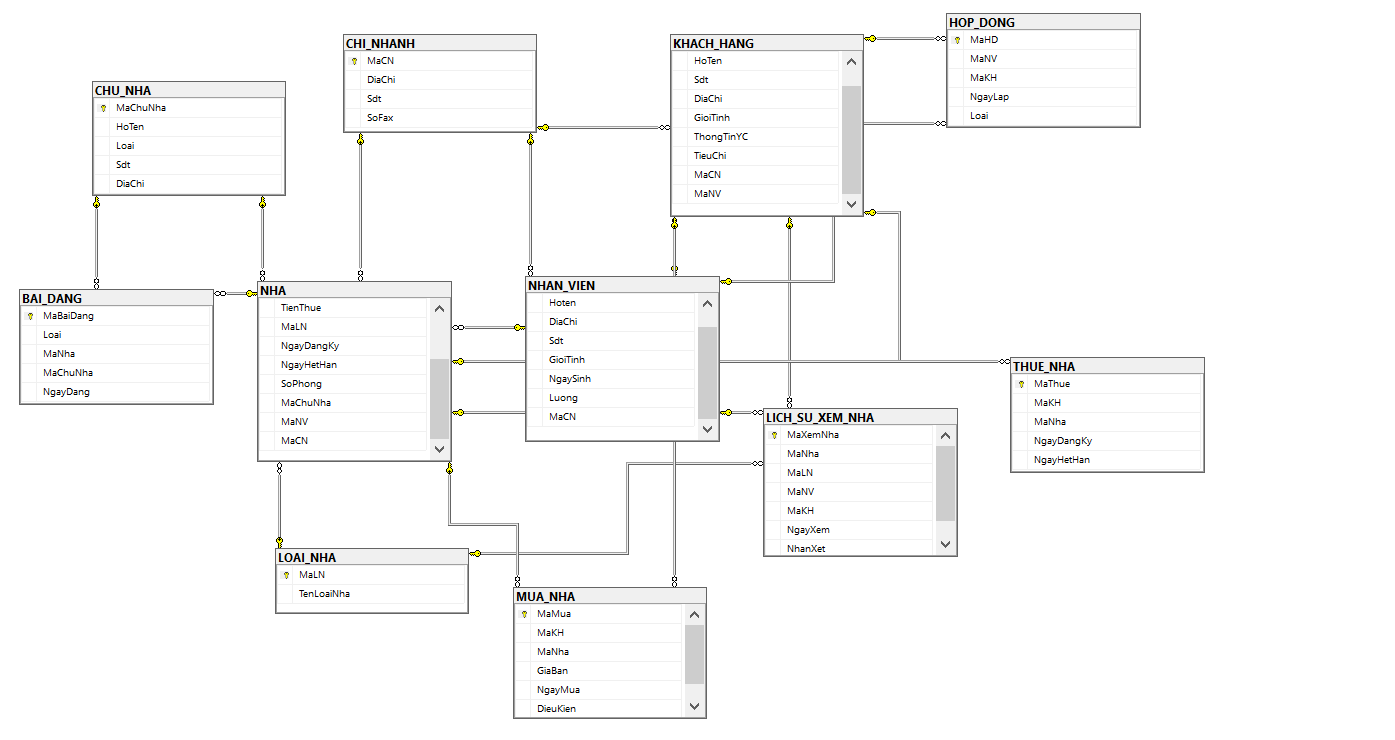
* 1. **R11: Tên của một loại nhà là duy nhất**

∀ loainha1, loainha2 ∈ LOAINHA ( loainha1 ≠ loainha2 ^ loainha1.tenLN ≠ loainha2.tenLN )

* 1. **R12: Nếu là nhà bán, giá bán và điều kiện chủ nhà không được để trống**
  2. **R13: Tiền lương nhân viên, tiền thuê nhà, bán nhà phải lớn hơn 0**
  3. **R14: Tiền mua nhà phải bằng tiền bán (sau khi xác nhận đã mua)**
  4. **R15: Số điện thoại nhân viên phải khác số điện thoại chi nhánh**

1. Cài đặt CSDL dùng SQL Server

* Diagram



* Table CHI\_NHANH

Ảnh có chứa bàn

Mô tả được tạo tự động

* Table NHAN\_VIEN

Ảnh có chứa bàn

Mô tả được tạo tự động

* Table CHU\_NHA

Ảnh có chứa bàn

Mô tả được tạo tự động

* Table NHA

Ảnh có chứa bàn

Mô tả được tạo tự động

* Table LOAI\_NHA



* Table KHACH\_HANG

Ảnh có chứa bàn

Mô tả được tạo tự động

* Table BAI\_DANG

Ảnh có chứa bàn

Mô tả được tạo tự động

* Table LICH\_SU\_XEM\_NHA

Ảnh có chứa bàn

Mô tả được tạo tự động

* Table HOP\_DONG

Ảnh có chứa bàn

Mô tả được tạo tự động

* Table THUE\_NHA

Ảnh có chứa bàn

Mô tả được tạo tự động

* Table MUA\_NHA

Ảnh có chứa bàn

Mô tả được tạo tự động

* Source code trong file script.

1. Xác định các loại người dùng
   * Khách hàng: quyền người dùng, chỉ được xem danh sách nhà, mua hay thuê nhà dựa vào hợp đồng được lập ra với nhân viên phụ trách.
   * Chủ nhà: quyền người dùng, chỉ được đăng bài cho thuê hay bán nhà thuộc quyền sở hữu của mình.
   * Nhân viên: quyền quản lý các khách hàng, tại mỗi chi nhánh sẽ có một trường chi nhánh quản lý toàn bộ nhân viên của chi nhánh đó.
2. Chức năng hệ thống: gồm ba loại người dung chính
   * Chức năng cho admin: quản lý (thêm, xóa, sửa, xem) danh sách nhân viên, chi nhánh, nhà, …
   * Chức năng cho nhân viên: quản lý danh sách nhà, các bản hợp đồng, thuê nhà, bán nhà hay lịch sử xem nhà của khách hàng.
   * Chủ nhà và khách hàng: chủ nhà có thể đăng bài cho thuê hay bán nhà, khách hàng có thể tìm kiếm nhà theo loại, thuê nhà hay mua nhà mà mình ưng ý.
3. Xác định lỗi tranh chấp
4. TH1

* Lost Update: Có hai giao tác đồng thời xảy ra, một khách hàng muốn mua một căn nhà, sau khi nhân viên A thay đổi giá nhà thành X thì bắt đầu lập hợp đồng cho khách hàng, tuy nhiên lúc đó nhân viên B đồng thời thay đổi giá nhà thành Y => Hợp đồng với giá tiền bị sai lệch => Lỗi lost update.

|  |  |
| --- | --- |
| T1 | T2 |
| Kiểm tra thông tin nhà A   * Set giá tiền mua nhà là X   Lập hợp đồng mua nhà với giá trị X  Commit | Kiểm tra thông tin nhà A   * Set giá trị nhà A từ giá X thành giá Y   Commit |

* Dirty Read: một nhân viên đang cập nhật trạng thái chưa thuê sang đã thuê của ngôi nhà, cùng lúc đó một nhân viên khác muốn kiểm tra tình trạng ngôi nhà đó thì thấy đã thuê, tuy nhiên vì ngôi nhà nhân viên cập nhật là nhà loại 4, mà nhà loại 4 chỉ cho bán nên hệ thống sẽ rollback về trạng thái chưa thuê=> Lỗi dirty read.

|  |  |
| --- | --- |
| T1 | T2 |
| Tìm nhà A và đọc thông tin nhà A   * Ghi nhà A sang trạng thái đã thuê   If@@error <> 0  Rollback  Commit | Tìm và đọc thông tin nhà A   * Thấy nhà A có trạng thái đã thuê   Commit |

* Unrepeatables Read: có hai giao tác đồng thời xảy ra, ban đầu nhân viên A kiểm tra giá thuê của nhà có mã là X sở hữu bởi chủ nhà Y, tuy nhiên đồng thời lúc đó có một nhân viên B thay đổi chủ sở hữu của nhà X thành Z nên sau đó nhân viên A xem lại giá tiền lần nữa thì không thấy giá tiền của nhà A => Lỗi unrepeatables read.

|  |  |
| --- | --- |
| T1 | T2 |
| Đọc: kiểm tra có nhà với mã nhà A thuộc sở hữu của Y hay không.  Đọc thông tin giá thuê nhà A với chủ sở hữa là Y (Không trả về đúng giá thuê đã kiểm tra trước đó)  Commit | Update nhà có mã nhà A thành nhà thuộc chủ nhà Z  Commit |

* Phantom Read: Một nhân viên A muốn kiểm tra xem khách hàng có mã là X đã mua bao nhiêu căn nhà, tuy nhiên cùng lúc đó nhân viên B đã thêm một ngôi nhà mà A vừa mua lúc đó, nhân viên A nhận được kết quả là N+1 ngôi nhà mặc dù trước đó chỉ thấy có N căn => Lỗi phantom read.

|  |  |
| --- | --- |
| T1 | T2 |
| Đọc: kiểm tra số nhà do khách hàng X sở hữu => set so\_nha = N  Đọc: thông tin của các ngôi nhà mà X sở hữu thì thấy có tới N + 1 ngôi nhà  Commit | Đọc: kiểm tra sự tồn tại của khách hàng X => Thêm một ngôi nhà vào danh sách ngôi nhà mà khách hàng X sở hữu  Commit |

1. TH2

* Lost Update: Có hai giao tác đồng thời xảy ra, chủ nhà X đổi căn nhà của mình từ loại nhà 1 sang loại nhà 2. Cùng lúc đó, nhân viên Y đến nhà của chủ nhà X thì thấy ngôi nhà đó thuộc loại nhà 3 nên đã cập nhật loại nhà của chủ nhà X từ 1 thành 3 => Những. giao tác sau này liên quan đến mua nhà và lập hợp đồng bị sai. => Lỗi lost update.

|  |  |
| --- | --- |
| T1 | T2 |
| Xem thông tin nhà X   * Set thông tin loại nhà X từ 1 thành 2   Xem thông tin loại nhà X thì thấy loại nhà X là 3 => Lỗi Lost Update  Commit | Xem thông tin nhà X   * Thay đổi thông tin loại nhà X từ 1 thành 3   Commit |

* Dirty Read: Tình huống: Một nhân viên (A) xem thông tin nhà X và cập nhật giá nhà X tử 3 tỷ lên 100 tỷ , cùng lúc đó một nhân viên (B) xem thông tin nhà X và thấy giá là 100 tỷ. Do hệ thống mua bán nhà đất không cho phép giao dịch quá 50 tỷ => Lỗi và phải rollback và quay lại trạng thái cũ với thông tin cũ là nhà X với giá 3 tỷ. =>Những gì nhân viên (B) đã nhìn thấy là không thật. => Lỗi Dirty Read

|  |  |
| --- | --- |
| T1 | T2 |
| Tìm nhà X và đọc thông tin nhà có giá Z.   * Đổi giá bán từ 3 tỷ thành 100 tỷ   If@@error <>0  Rollback  Commit | Tìm kiêm thông tin nhà X và đọc thông tin.   * Thấy nhà X có giá bán 100 tỷ   Commit |

* Unrepeatable Read: Có hai giao tác đồng thời xảy ra. Quản trị viên (A) đang kiểm tra thông tin số điện thoại khách hàng, thsố điện thoại là X, trong khi đó một nhân viên (B) đã cập nhật thông tin liên lạc của khách hàng đó từ X thành Y. Làm cho nhân viên (A) kiểm tra và đọc lại thông tin bị lỗi và không thấy số điện thoại ban đầu. => Lỗi Unrepeatable Read

|  |  |
| --- | --- |
| T1 | T2 |
| Quản trị viên đọc, kiểm tra và thấy số điện thoại của khách hàng là X  Quản trị viên đọc lại thông tin khách hàng:   * Không thấy thông tin số điện thoại cũ đâu, thay vào đó thì thấy số điện thoại Y   Commit | Nhân viên đọc số điện thoại và cập nhật lại số điện thoại của khách hàng từ X thành Y  Commit |

* Phantom Read: Quản trị viên (A) đang xem danh sách hợp đồng với số lượng hợp đồng là N, tuy nhiên cùng lúc đó nhân viên (B) thêm mới một hợp đồng mua nhà vào danh sách hợp đồng. Kết quả khi quản trị viên đọc lại danh sách hợp đồng thì lại thấy số lượng hợp đồng là N+1. => Lỗi Phantom Read

|  |  |
| --- | --- |
| T1 | T2 |
| Đọc: kiểm tra số lượng hợp đồng hiện có => set so\_hop\_dong = N  Đọc: kiểm tra lại thấy số lượng hợp đồng trong danh sách có tới N+1 hợp đồng  Commit | Đọc: xem danh sách hợp đồng và thành lập một hợp đồng mua nhà vào danh sách hợp đồng.  Commit |

1. TH3

* Lost Update: chủ của ngôi nhà A muốn gia hạn thêm ngày hết hạn là 9 tháng, nhân viên X nhận được yêu cầu, tuy nhiên sau đó chủ nhà muốn gia hạn thêm 3 tháng nữa để tròn 12 tháng tuy nhiên lúc này nhân viên B lại nhận yêu cầu, vào cùng một thời điểm cả hai nhân viên đều gia hạn cho ngôi nhà nhưng kết quả thời gian gia hạn lại không đồng nhất với mong muốn của chủ nhà => lỗi lost update.

|  |  |
| --- | --- |
| T1 | T2 |
| Đọc thông tin nhà A   * Kiểm tra ngày đăng ký và ngày hết hạn của nhà A   Update ngày hết hạn mới là X  Commit | Đọc thông tin nhà A   * Kiểm tra ngày đăng ký và ngày hết hạn của nhà A   Update ngày hết hạn mới là Y  Commit |

* Dirty Read: nhân viên A cập nhật nhà X từ N phòng lên N+1 phòng, lúc đó nhân viên B đang kiểm tra số phòng từng nhà thì thấy nhà X có N + 1 phòng, tuy nhiên mỗi nhà chỉ được phép chứa tối đa 30 phòng mà N+1 lúc này lớn hơn 30 nên hệ thống rollback và khôi phục về N phòng của nhà X => lỗi dirty read.

|  |  |
| --- | --- |
| T1 | T2 |
| Nhân viên: Tìm nhà có mã nhà là X và thấy nhà có N phòng   * Set số phòng = N + 1   If@@error <> 0  Rollback(khôi phục lại N phòng)  Commit | Khách hàng: Tìm và đọc thông tin nhà X   * Thấy nhà X có N+1 phòng   Commit |

* Unrepeatable Read: nhân viên A muốn kiểm tra thông tin của hợp đồng X mà nhân viên Y đã lập, tuy nhiên lúc đó nhân viên B đã thay đổi người lập hợp đồng X là nhân viên Z nên nhân viên A không thấy được thông tin sau khi truy vấn => Lỗi unrepeatables read.

|  |  |
| --- | --- |
| T1 | T2 |
| Đọc: kiểm tra có hợp đồng X được lập bởi nhân viên Y hay không  Đọc thông tin hợp đồng X với mã nhân viên là Y(không hiển thị thông tin do lỗi)  Commit | Update hợp đồng X từ mã nhân viên Y sang mã nhân viên Z  Commit |

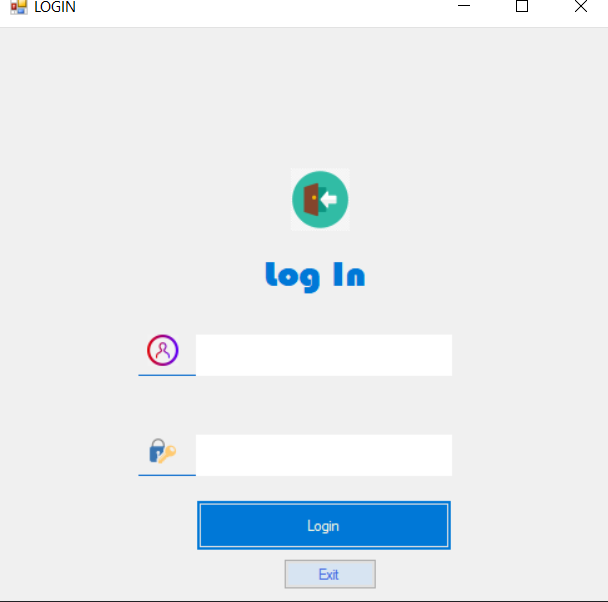
* Phantom Read: nhân viên muốn kiểm tra xem khách hàng A đã thuê bao nhiêu căn nhà thì thây N căn, tuy nhiên đồng thời lúc đó nhân viên khác đã thêm vào một căn nhà thuê mà khách hàng A đã thuê, nên sau đó nhân viên kia lại thấy khách hàng A thuê N+1 căn nhà => Lỗi Phantom Read.

|  |  |
| --- | --- |
| T1 | T2 |
| Kiểm tra số nhà đã thuê của khách hàng A   * Set so\_nha = N   Đọc thấy có N + 1 nhà đã thuê của khách hàng A(thêm một ngôi nhà)  Commit | Đọc: Kiểm tra thông tin khách hàng A, thêm vào 1 nhà đã thuê vào danh sách nhà đã thuê của khách hàng A   * Có N +1 nhà thuê của khách hàng A   Commit |

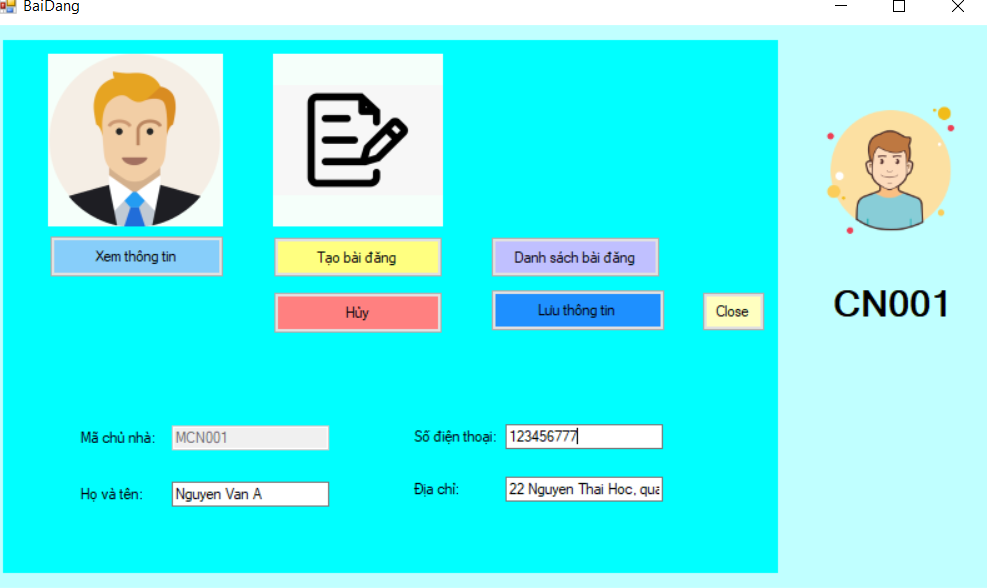
1. Thiết kế giao diện

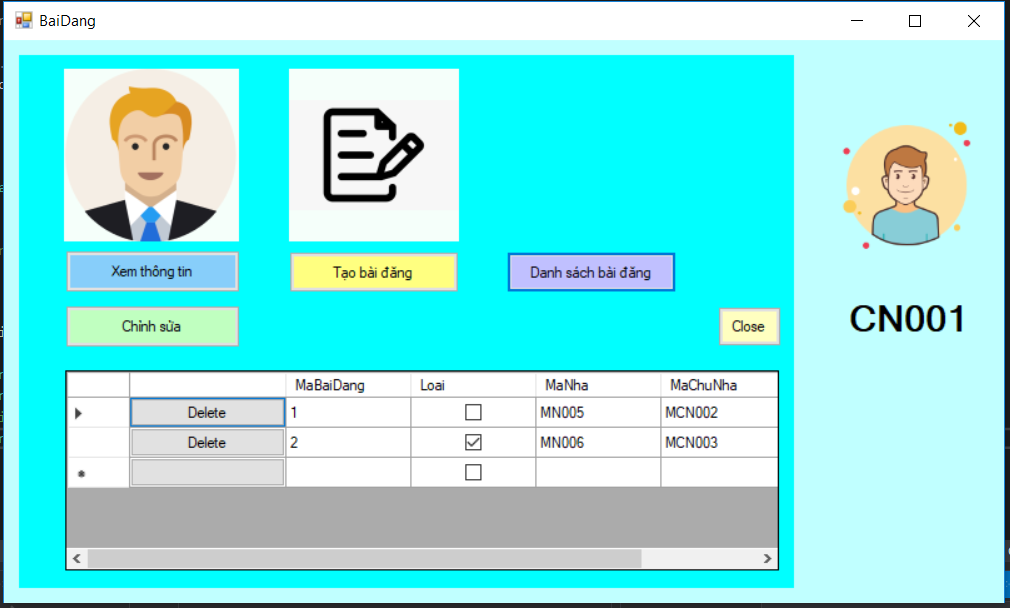
* Giao diện trang chủ và login

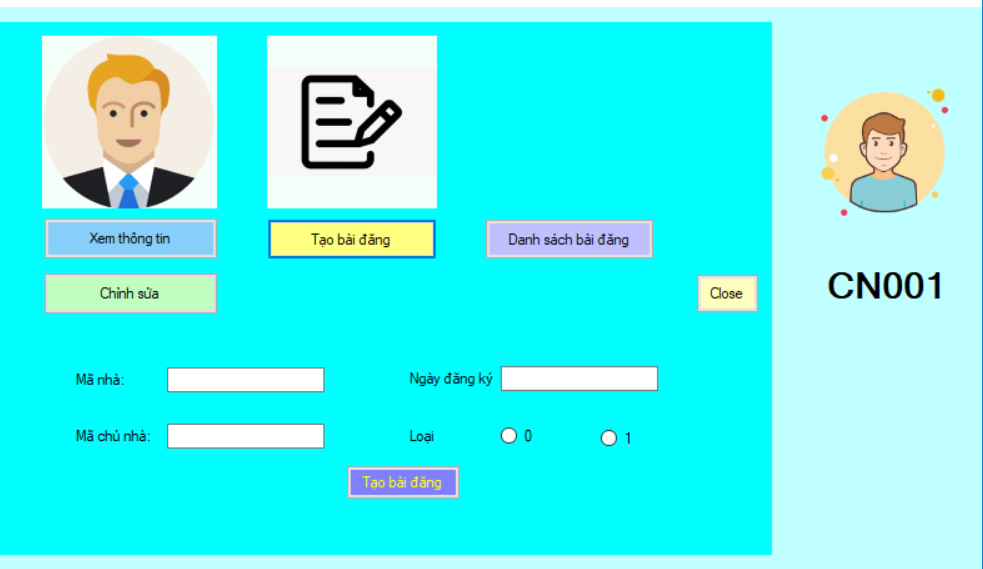
****

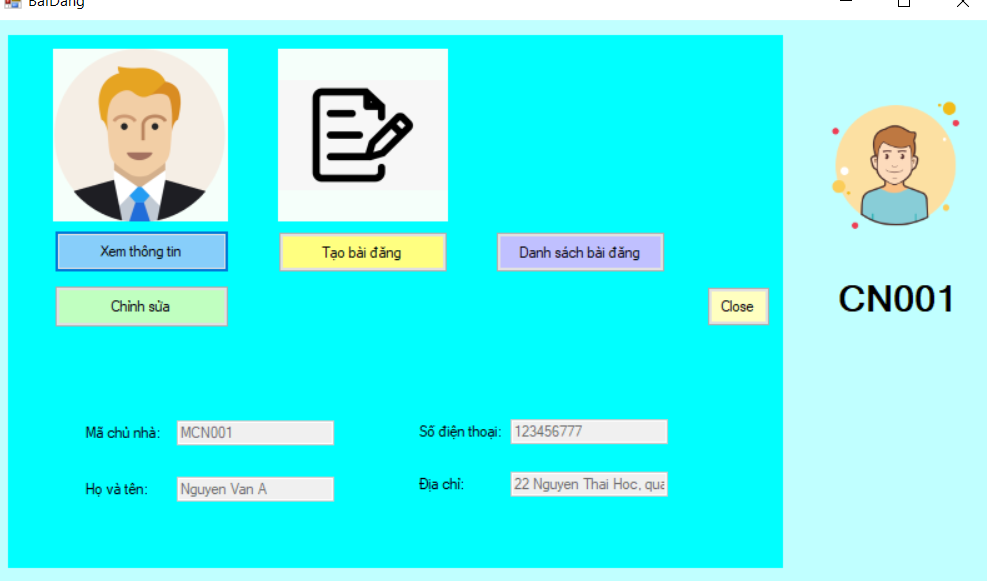
****

* Giao diện chủ nhà

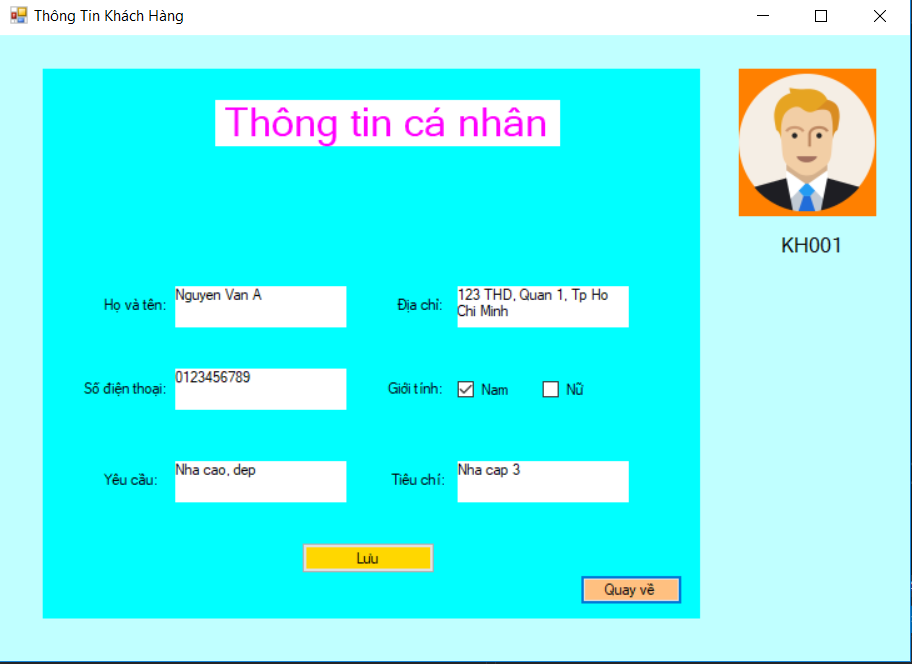
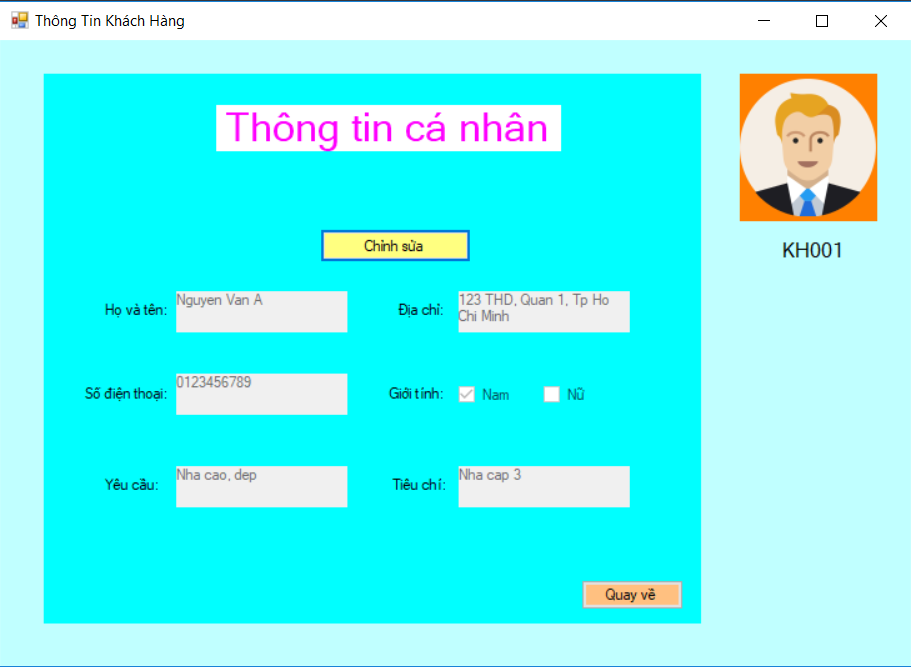
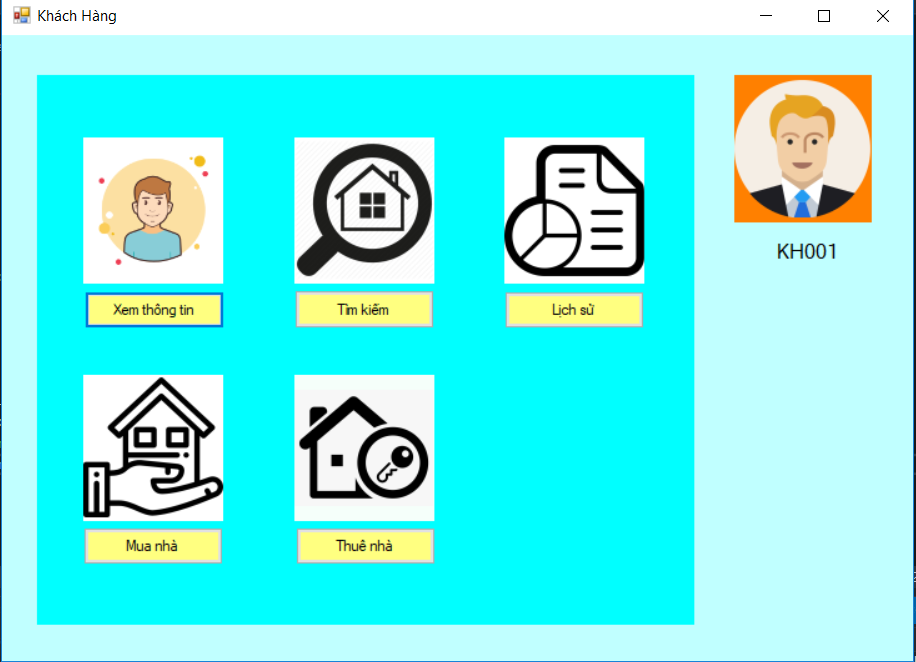


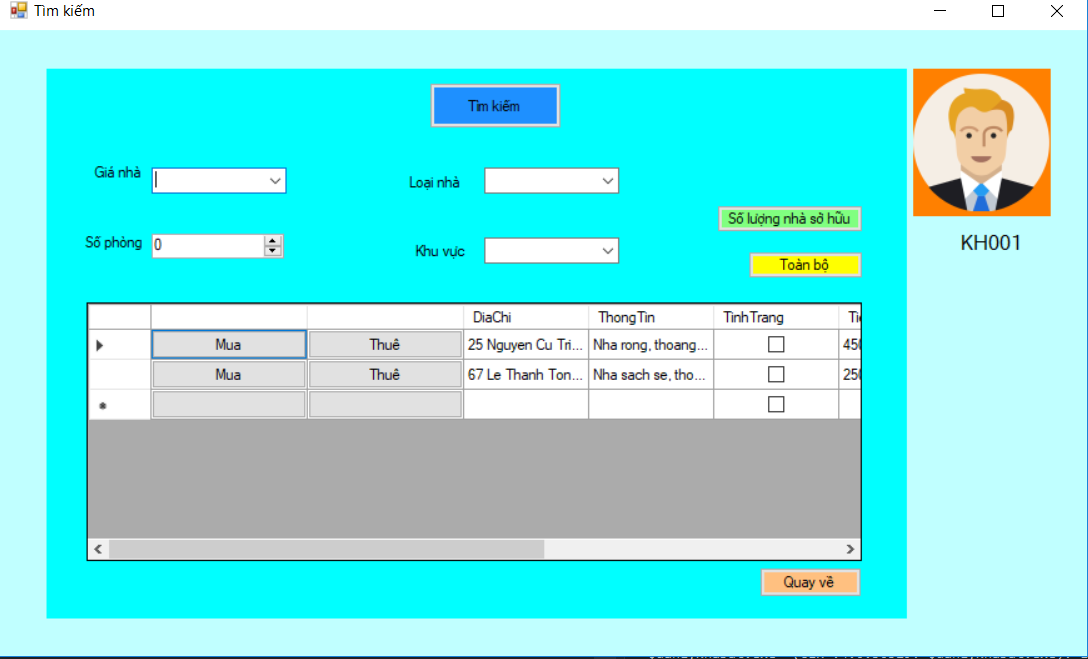




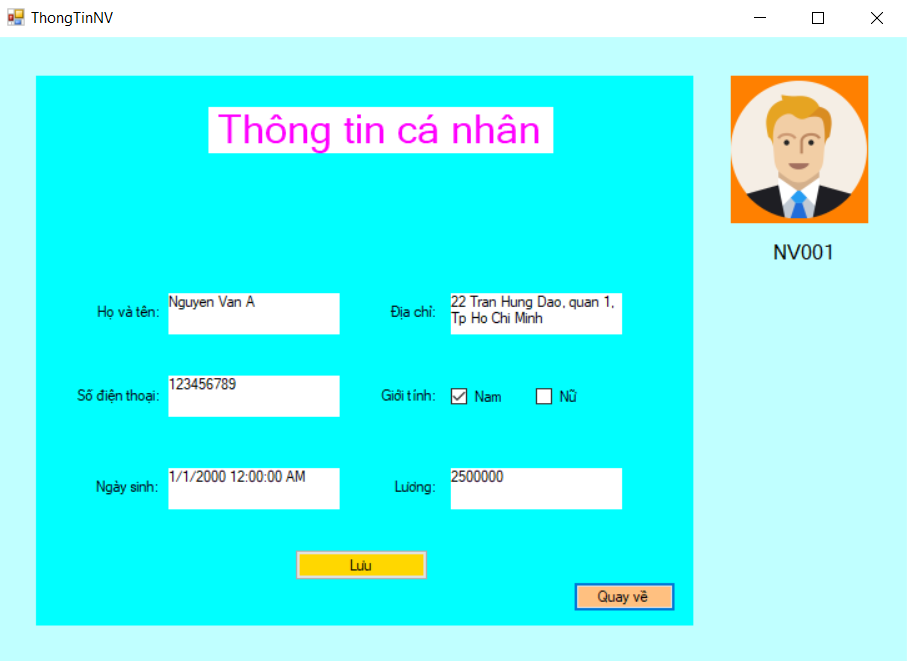
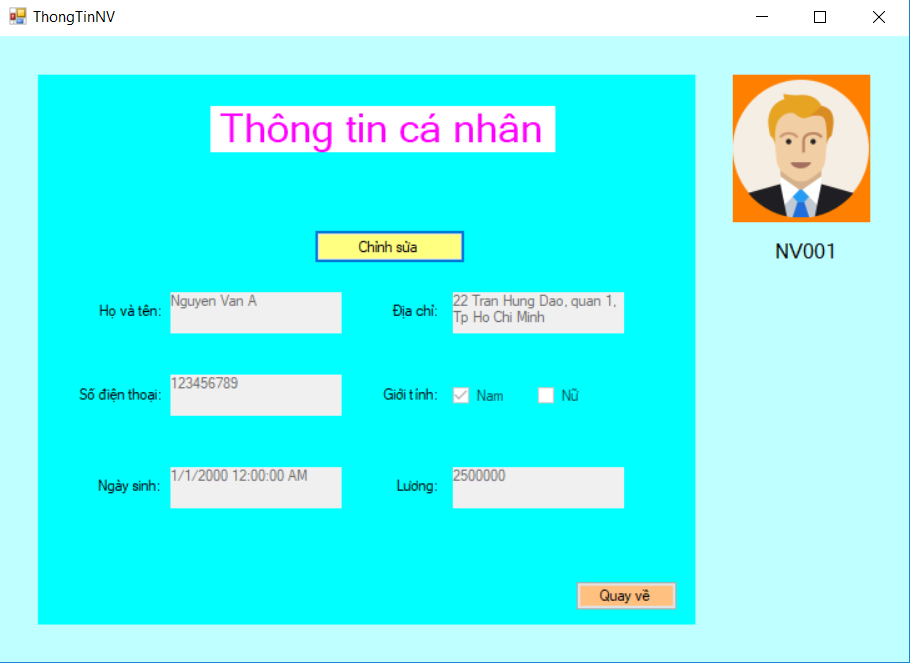
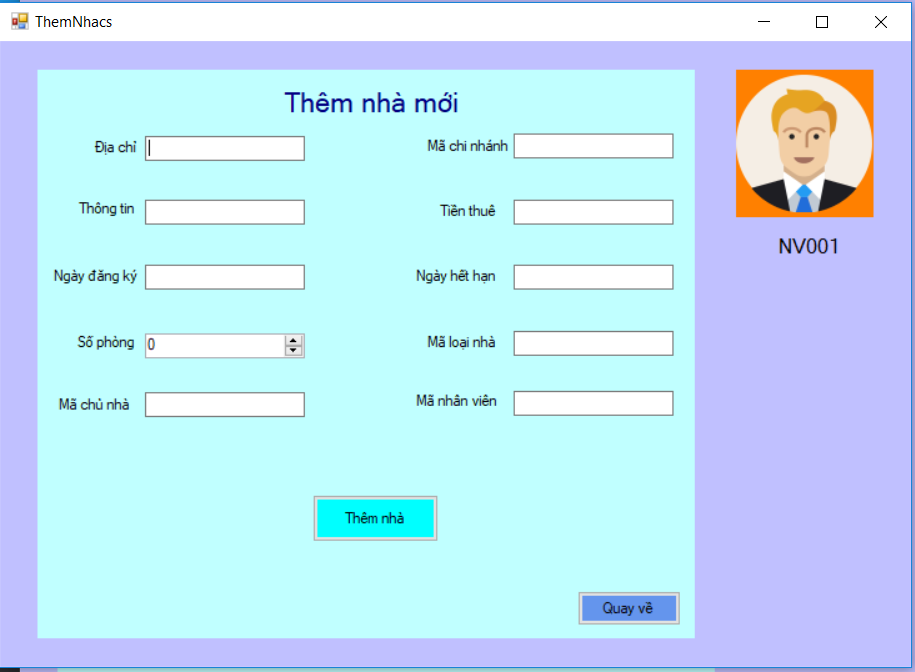
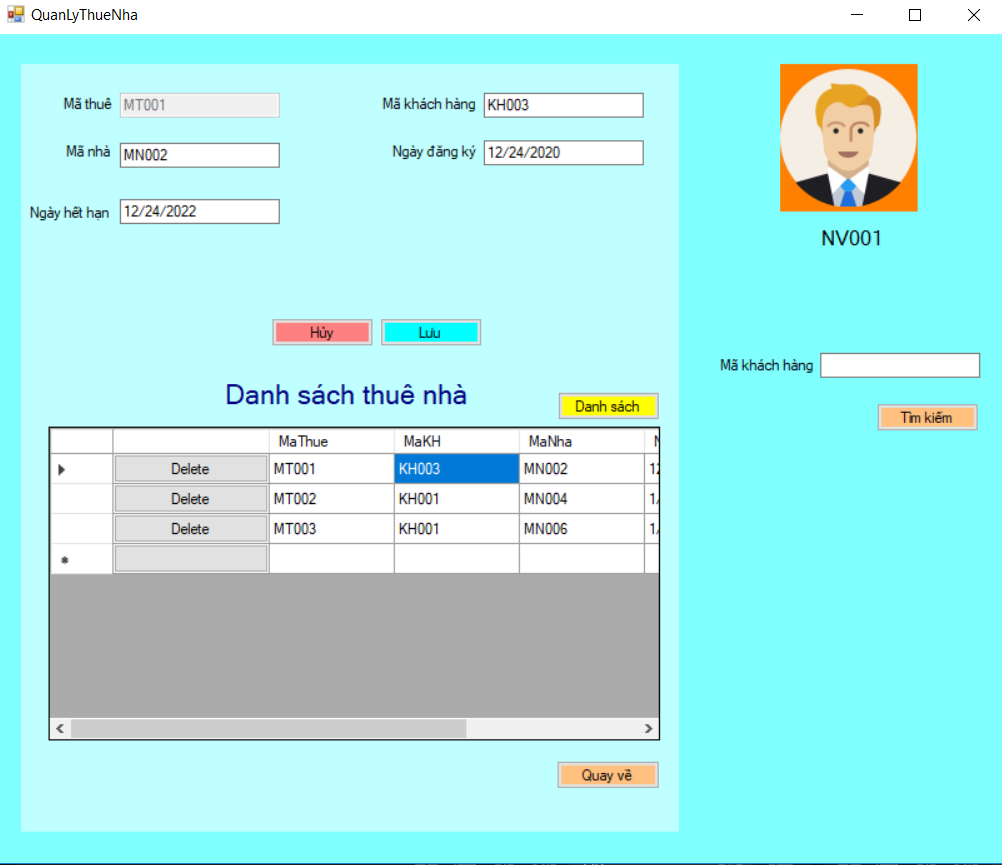
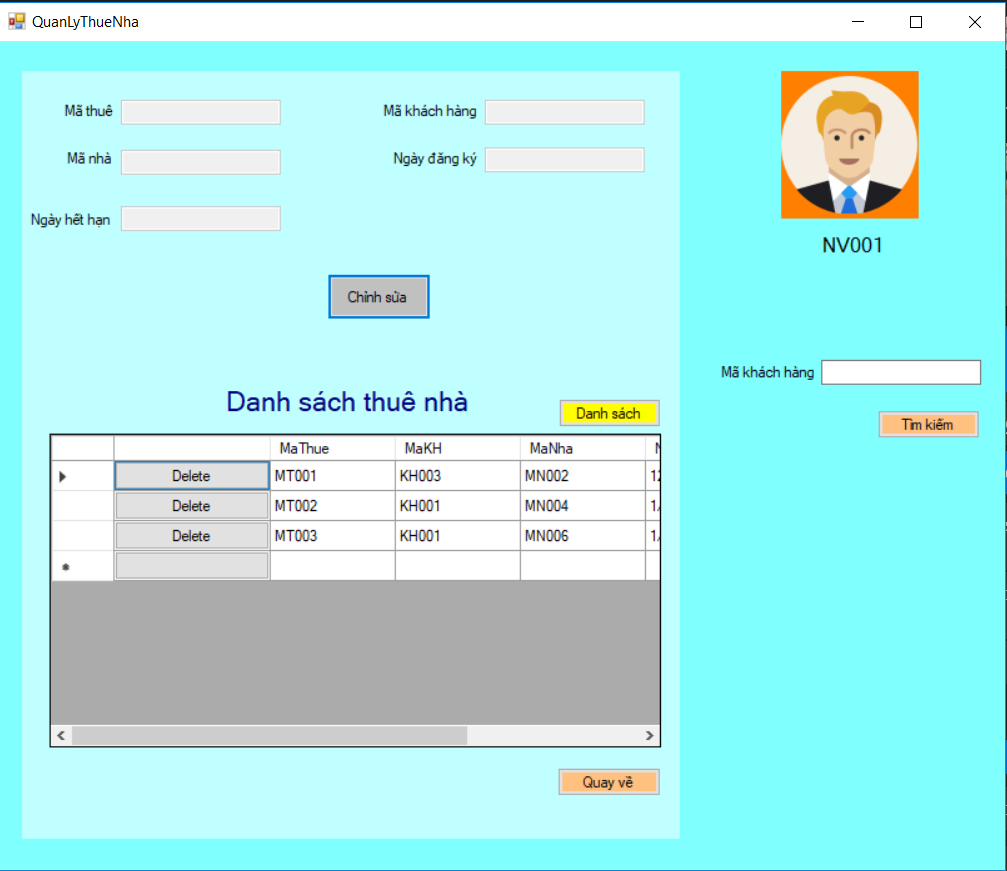
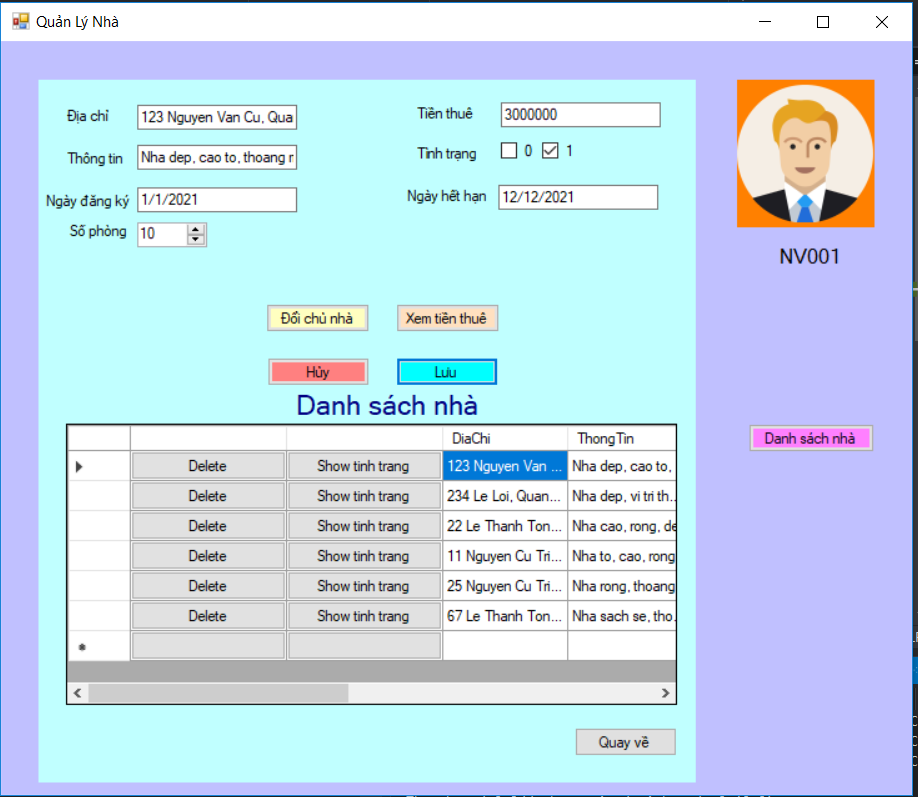
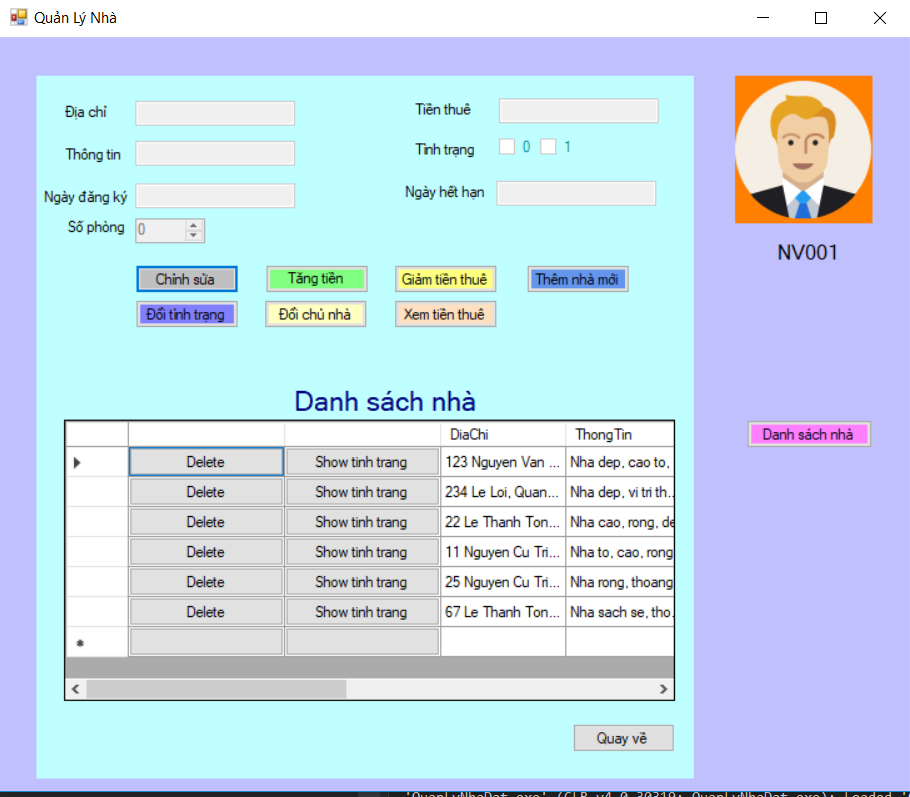
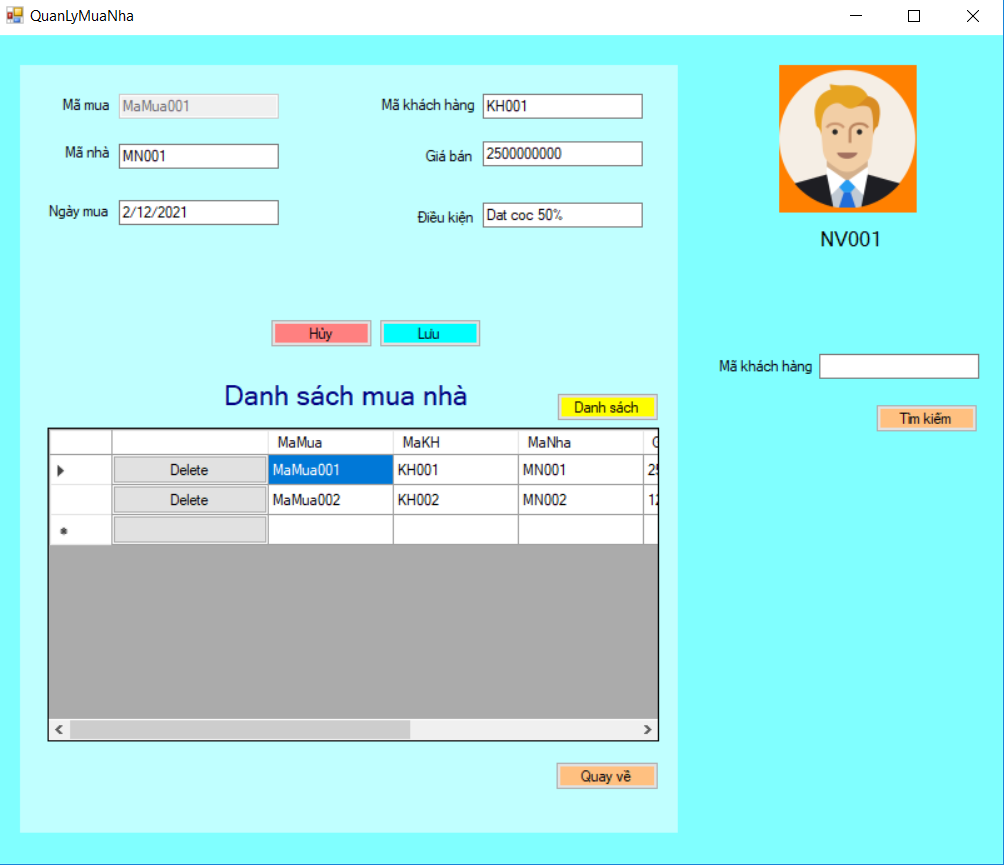
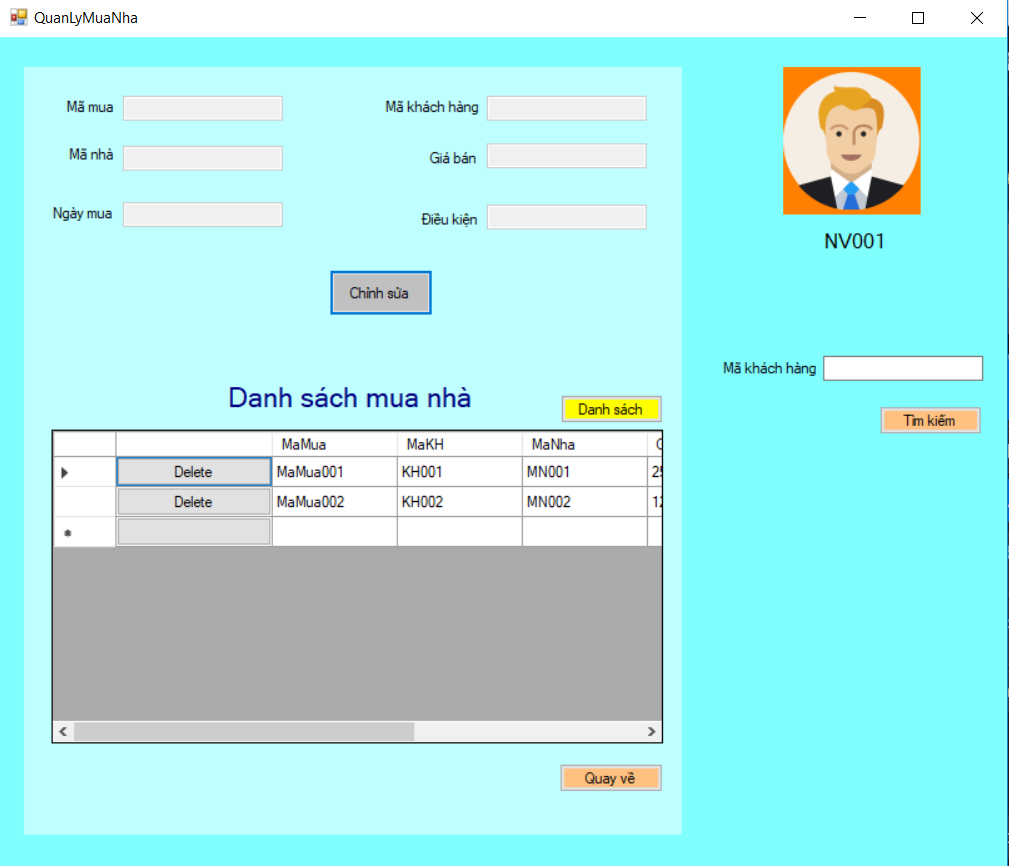
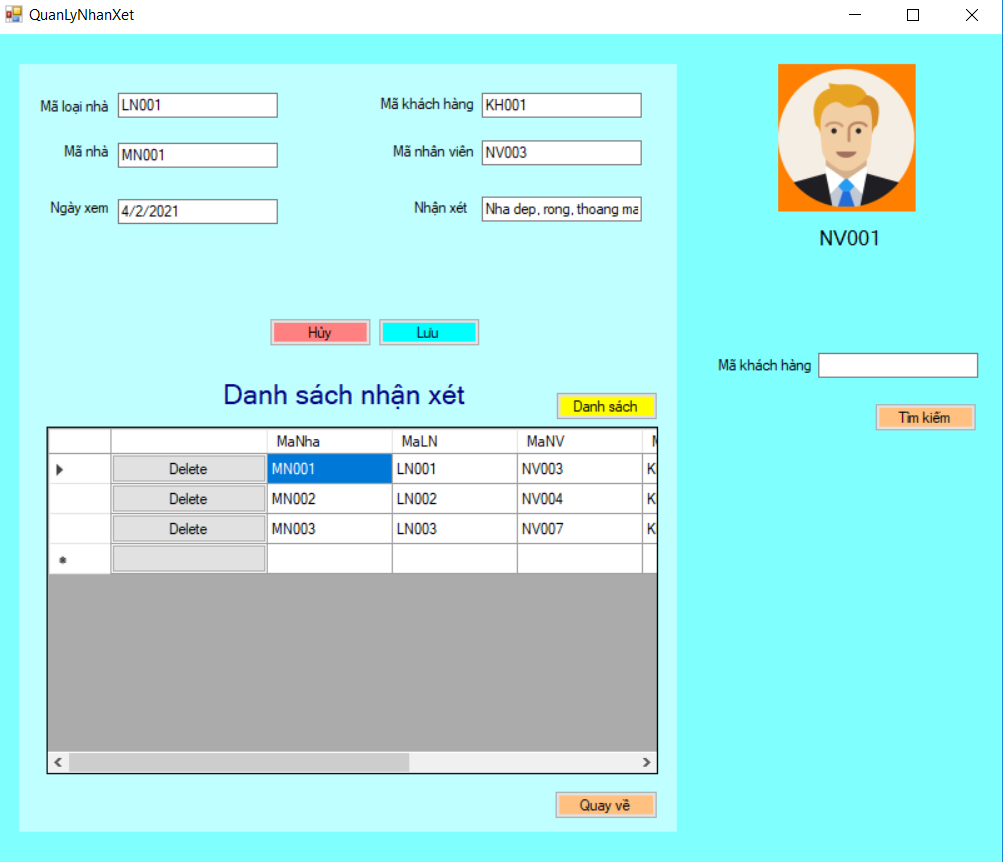
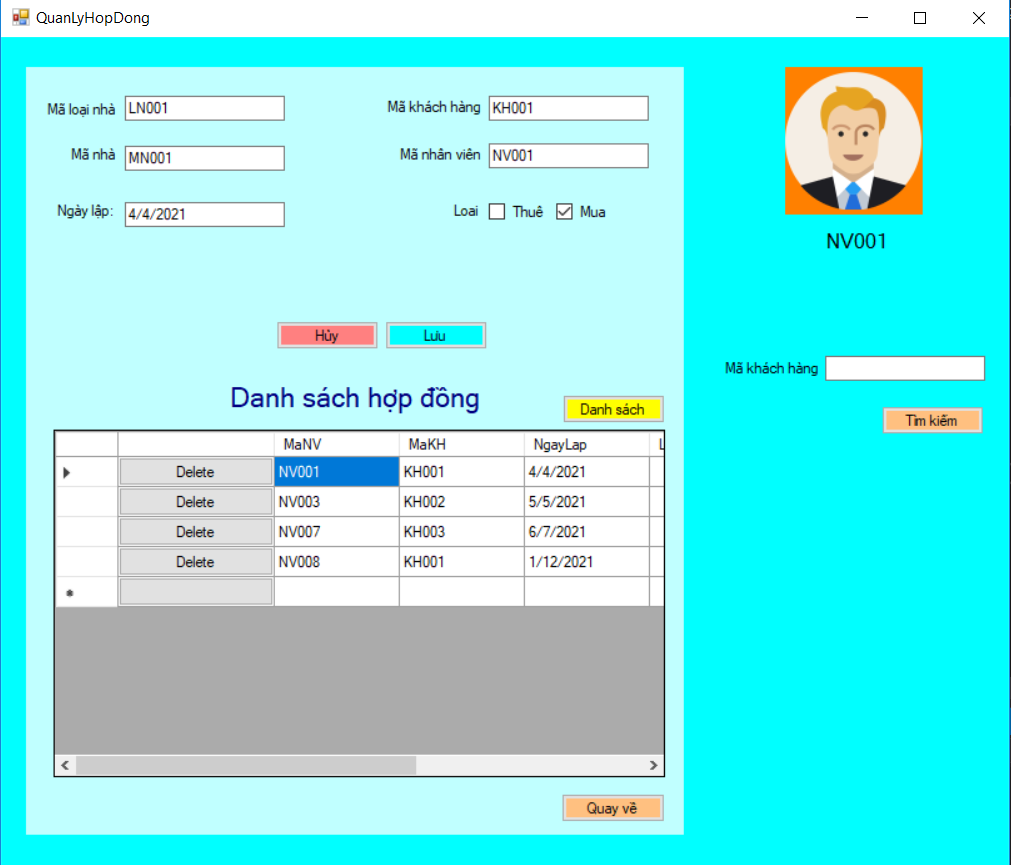


* Giao diện khách hàng





* Giao diện nhân viên



1. Giao tác chức năng hệ thống

Ảnh có chứa văn bản

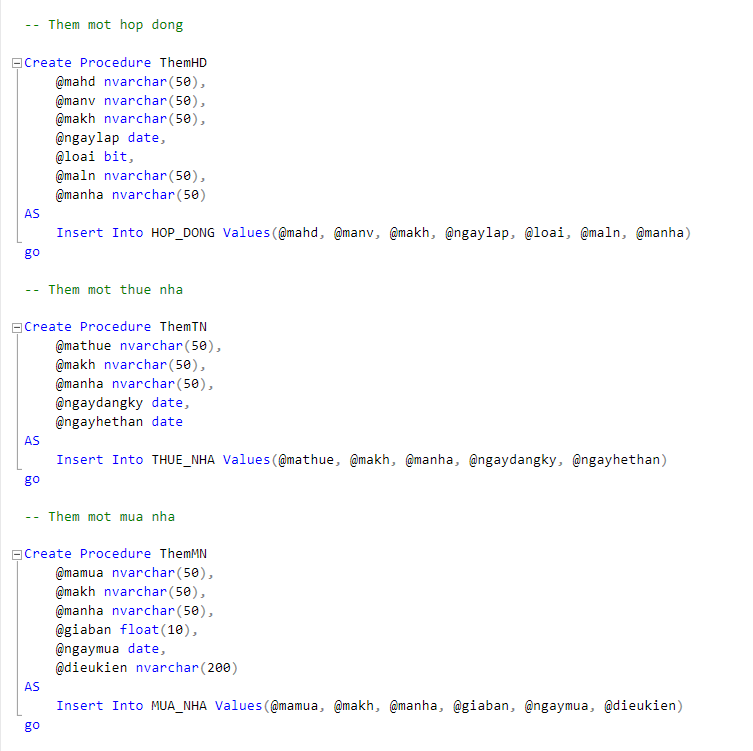
Mô tả được tạo tự động

Ảnh có chứa văn bản

Mô tả được tạo tự động

Ảnh có chứa văn bản

Mô tả được tạo tự động



Ảnh có chứa văn bản

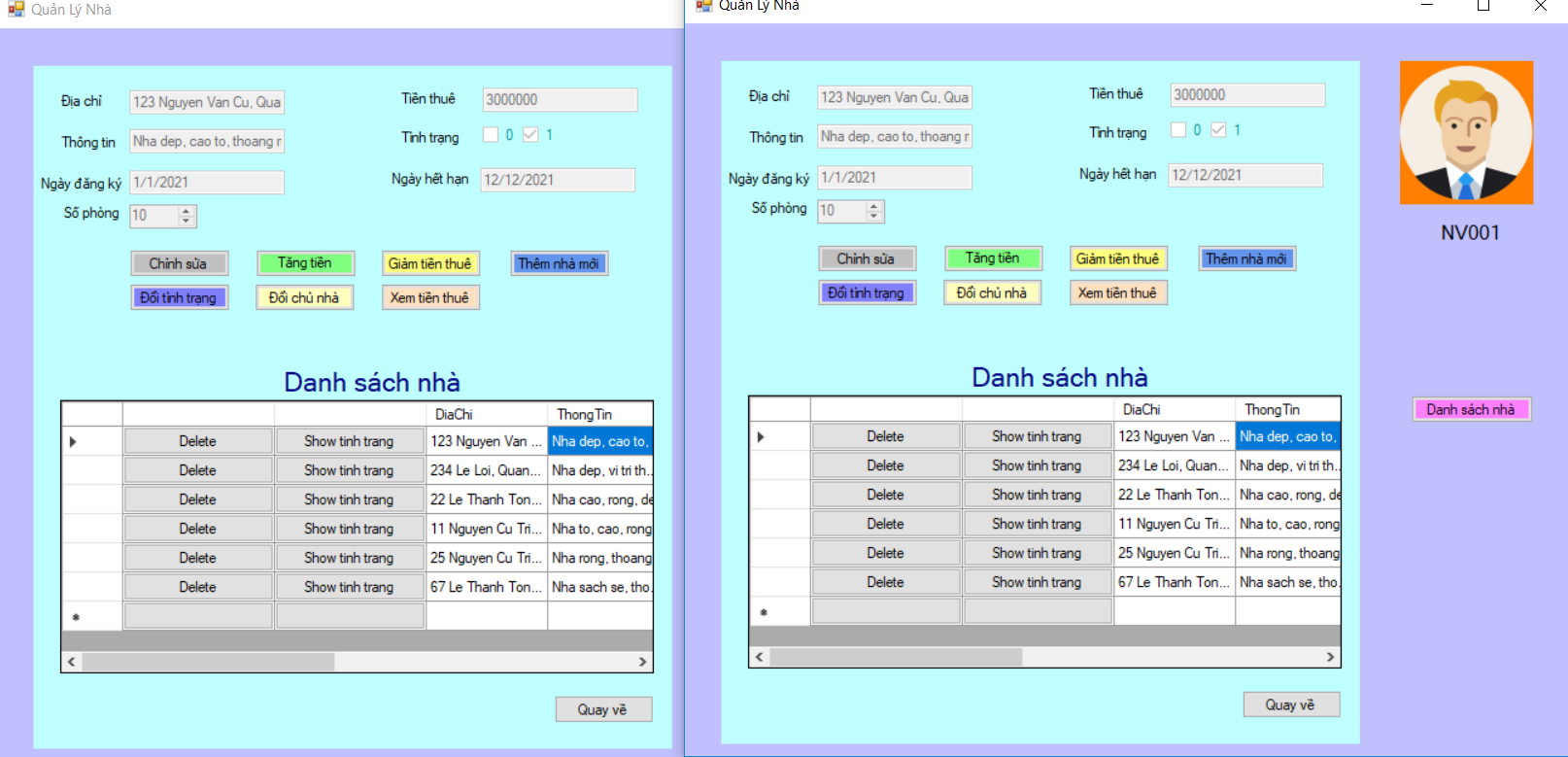
Mô tả được tạo tự động

1. Xác định tranh chấp đồng thời, xử lý lỗi và demo lỗi
2. TH1

* Lost Update: T1 và T2 thực hiện với mức cô lập RC

|  |  |  |  |
| --- | --- | --- | --- |
| T1 | Khóa | T2 | Khóa |
| If Not Exists (Select \* From NHA Where MaNha = @manha)  Begin  Print 'Khong tim thay doi tuong'  RollBack Transaction  End  Select @TienThue = TienThue  From NHA  Where MaNha = @manha | T1 xin khóa S  Hệ thống  Phát khóa S |  |  |
|  |  | If Not Exists (Select \* From NHA Where MaNha = @manha)  Begin  Print 'Khong tim thay doi tuong'  RollBack Transaction  End  Select @TienThue = TienThue  From NHA Where  MaNha = @manha | T2 xin khóa S  Hệ thống phát  khóa S |
| Set @TienThue = @TienThue + 1000000  Update NHA  Set TienThue = @TienThue  Where MaNha = @manha | T1 xin khóa X,  do mức cô lập RC  nên hệ thống phát khóa X cho T1 |  |  |
|  |  | Set @TienThue = @TienThue - 500000  Update NHA  Set TienThue = @TienThue  Where MaNha = @manha | Do T1 đang giữ  khóa X nên hệ  thống sẽ không phát khóa X cho T2  T2 phải đợi T1 commit |
| Commit |  |  |  |
|  |  | Commit |  |

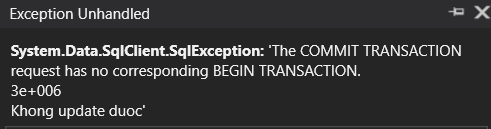
* + - * + Thiết lập mức cô lập Repeatable Read cho cả hai giao tác T để xử lý lỗi Lost Update.
      * Demo lỗi



Ban đầu cả hai bảng đều có số tiền là 3000000



Kết quả tiền thuê ở mỗi bên khác nhau sau khi một bên tăng tiền còn một bên giảm tiền.

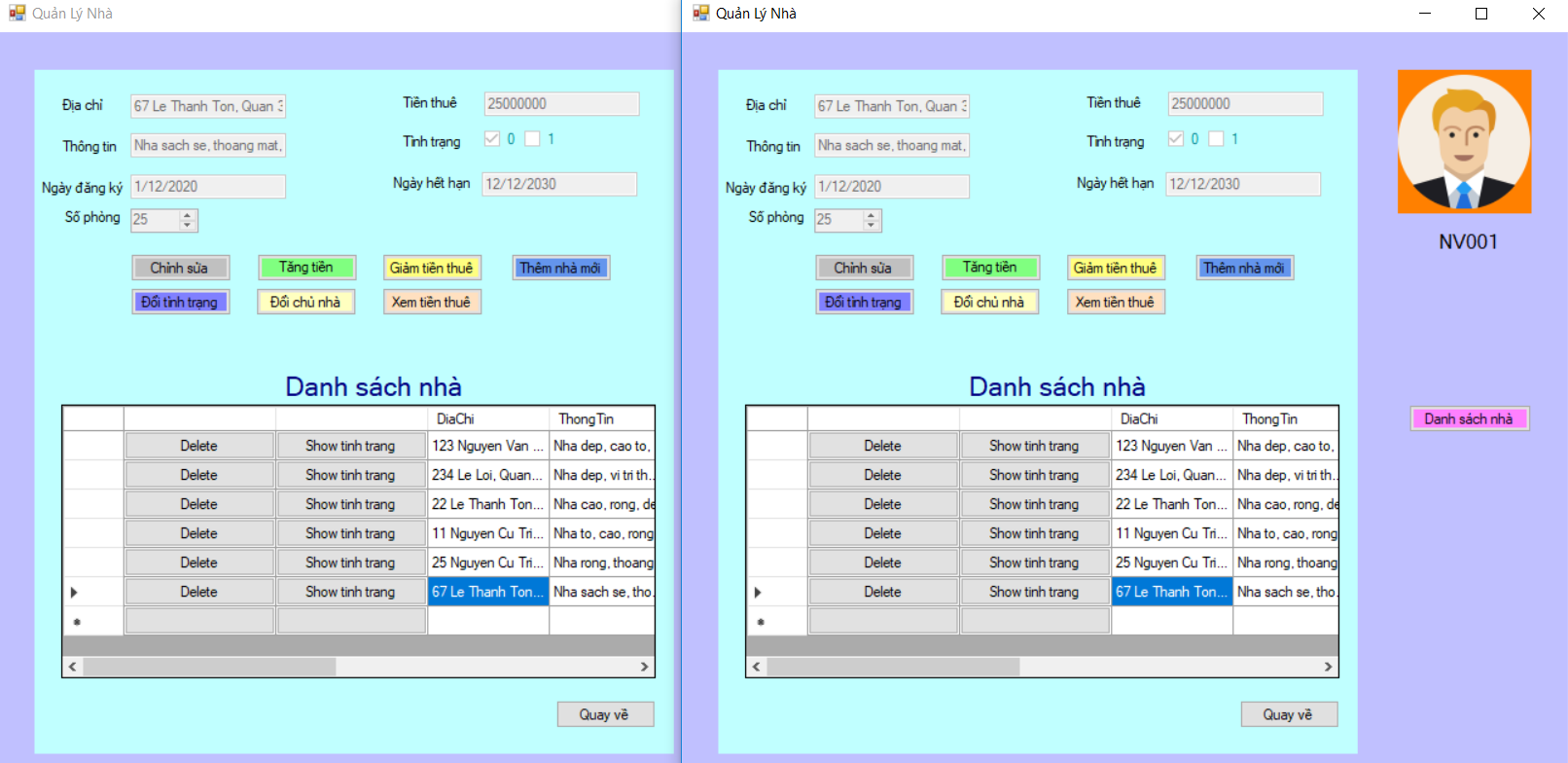


Kết quả sau khi xử lý lỗi lost update: hệ thống bị deadlock nên một bên sẽ bị dừng lại.

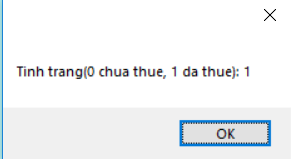
* Dirty Read: T1 với mức cô lập RC, T2 với mức cô lập URC

|  |  |  |  |
| --- | --- | --- | --- |
| T1 | Khóa | T2 | Khóa |
| If Not Exists (Select \* From NHA Where MaNha = @manha)  Begin  Print 'Khong tim thay doi tuong'  End  Select @maln = MaLN  From NHA  Where MaNha = @manha | T1 xin khóa  S  Hệ thống  phát khóa S |  |  |
| Update NHA set  TinhTrang = 1  Where MaNha = @manha | T1 xin khóa X. Hệ thống  phátkhóa X |  |  |
|  |  | Set Transaction Isolation Level Read UnCommitted  Select TinhTrang  From NHA Where  MaNha = @manha | T1 đang giữ khóa X  Tuy nhiên do T2 có mức  cô lập URC nên T2 không  cần xin khóa. |
| If @@error <>0  Rollback Transaction |  | Commit |  |
| Commit |  |  |  |

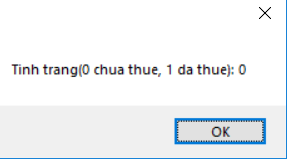
* + - Thiết lập mức cô lập Read Committed cho T2 để xử lý lỗi Dirty Read.
      * Demo lỗi



Ban đầu cả hai bên đều nhận được tình trạng của ngôi nhà thứ 6 là 0 (chưa thuê)



Khi thay đổi tình trạng nhà thứ 6, vì nhà này thuộc loại LN004 ( chỉ được bán không cho thuê) nên không thể đổi sang tình trạng 1(đã thuê) được mà hệ thống sẽ rollback và trả về tình trạng 0 cho nhà thứ 6. Nhưng do show tình trạng đọc với mức URC nên vẫn đọc cái đã thuê của T1.



Kết quả show tình trạng khi nhấn lần 2

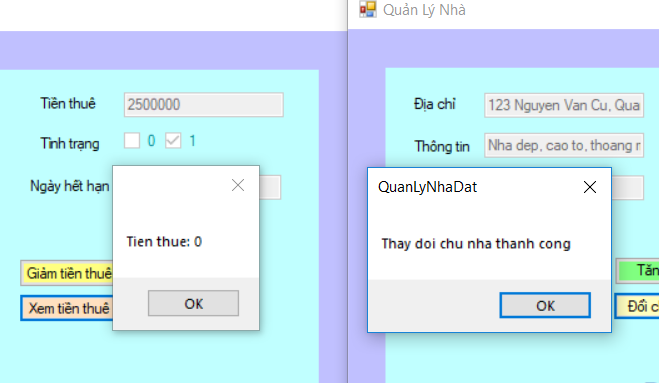


Kết quả đúng như mong đợi sau khi thiết lập mức cô lập RC cho T2, nhà thứ 6 sau khi ấn thay đổi tình trạng thì tình trạng vẫn không thay đổi và bên T2 đọc được tình trạng vẫn là 0 (chưa thuê) sau khi ấn vô show tình trạng.

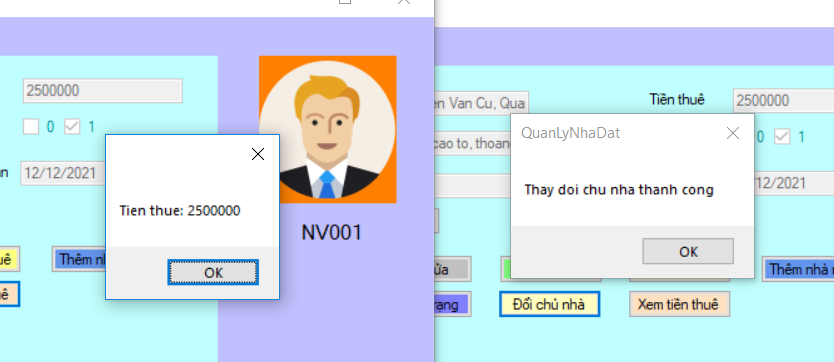
* Unrepeatables Read: T1 và T2 với mức cô lập RC

|  |  |  |  |
| --- | --- | --- | --- |
| T1 | Khóa | T2 | Khóa |
| If Not Exists  (Select \* From NHA  Where MaNha = @manha) | T1 xin khóa S  Hệ thống phát  khóa S |  |  |
| Select @machunha = MaChuNha  From NHA  Where MaNHa = @manha | T1 xin khóa S  Hệ thống phát  khóa S |  |  |
|  |  | Update NHA  Set MaChuNha = @machunha  Where MaNha = @manha | T2 xin khóa X  Hệ thống phát  khóa X |
| Select \* From NHA  Where MaNha = @manha and  MaChuNha = @machunha | T1 xin khóa S  Vì T2 đang giữ  khóa X nên không  xin được khóa. Phải  đợi T2 commit mới  xin được khóa |  |  |
| Commit |  | Commit |  |

* + - Thiết lập mức cô lập Repeatable Read cho T1 để xử lý lỗi Unrepeatable Read.
      * Demo lỗi



Lỗi unrepeables read khi một bên kiểm tra tiền thuê đồng thời bên kia update mã chủ nhà của ngôi nhà mà nhân viên kiểm tra đang truy vấn

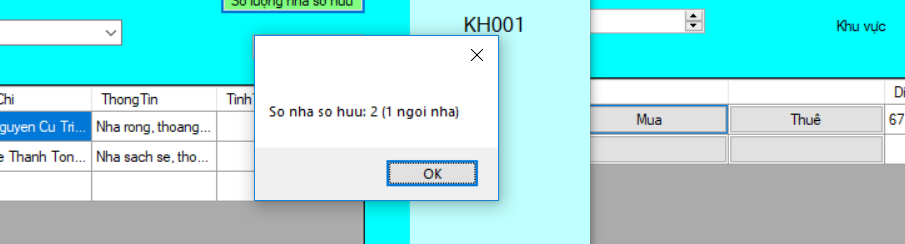


Sau khi thiết lập mức cô lập Repeatables Read cho T1, hệ thống đã chạy bình thường mà không có lỗi

* Phantom Read:

|  |  |  |  |
| --- | --- | --- | --- |
| T1 | Khóa | T2 | Khóa |
| Select @nb = count(\*)  From MUA\_NHA  Where MaKH = @makh  Print @nb | T1 xin khóa S  Hệ thống phát khóa  S |  |  |
|  |  | If Exists (Select \*  From MUA\_NHA  Where MaMua = @mamua)  Print 'Da ton tai  doi tuong'  Else  Insert Into MUA\_NHA  Values(@mamua, @makh,  @manha,@tien,@ngay,  @dieukien) | T2 xin khóa S Sau đó xin khóa X. Hệ thống phát khóa X cho T2 |
| Select @nb1 = count(\*) From MUA\_NHA  Where MaKH = @makh  Print @nb1 | T1 xin khóa S, nhưng  do T2 đang giữ khóa  X nên T1 phải chờ T2 commit |  |  |
|  |  | Commit |  |
| Commit |  |  |  |

* + - Thiết lập mức cô lập Serializable cho T1 để xử lý lỗi Phantom Read.
      * Demo lỗi



Khi xem số nhà sở hữu thì nhận thấy có thêm 1 ngôi nhà do lúc đó đã có một giao tác mua nhà khác chạy đồng thời



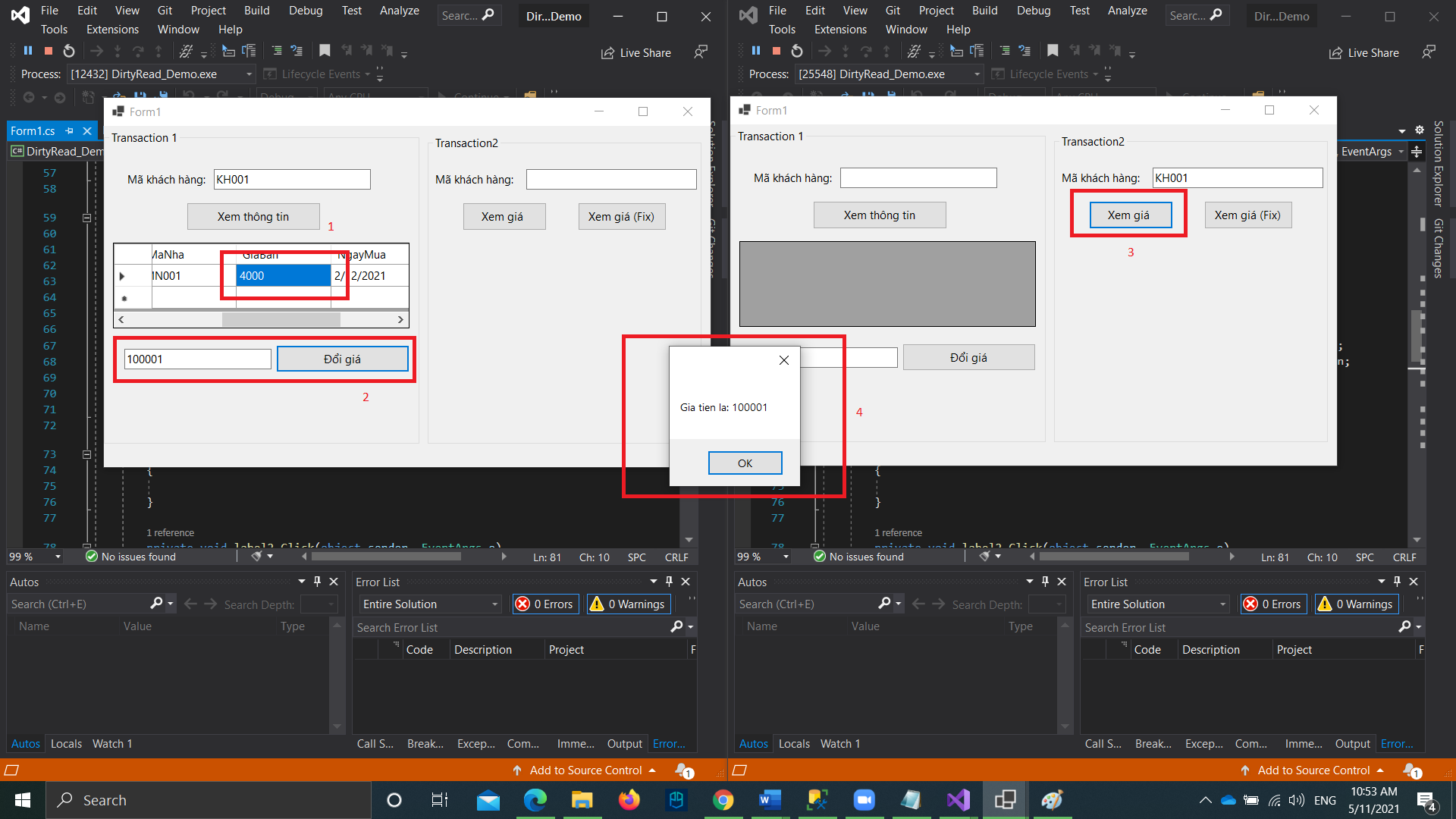
Sau khi thiết lập mức cô lập Serializable cho T1 thì hệ thống đã chạy như bình thường.

1. TH2:

* Lost Update: T1 và T2 thực hiện với mức cô lập RC

|  |  |  |  |
| --- | --- | --- | --- |
| T1 | Khóa | T2 | Khóa |
| Select \*  From NHA  Where MaNha | T1 xin khóa S  Hệ thống  Phát khóa S |  |  |
|  |  | Select \*  From NHA  Where MaNha | T2 xin khóa S  Hệ thống phát  khóa S |
| Update NHA  Set MaLN = @maln  Where MaNha = @manha;  Declare @maln\_moi nvarchar(50)  Select @maln\_moi = MaLN  From NHA Where MaNha = @manha;  UPDATE HOP\_DONG  Set MaLN = @maln  Where MaNha = @manha; | T1 xin khóa X,  do mức cô lập RC  nên hệ thống phát khóa X cho T1 |  |  |
|  |  | UPDATE NHA  Set MaLN = @maln  where MaNha = @manha  Declare @maln\_moi nvarchar(50)  Select @maln\_moi = MaLN  From NHA  Where MaNha = @manha; | Do T1 đang giữ  khóa X nên hệ  thống sẽ không phát khóa X cho T2  T2 phải đợi T1 commit |
| Commit |  |  |  |
|  |  | Commit |  |

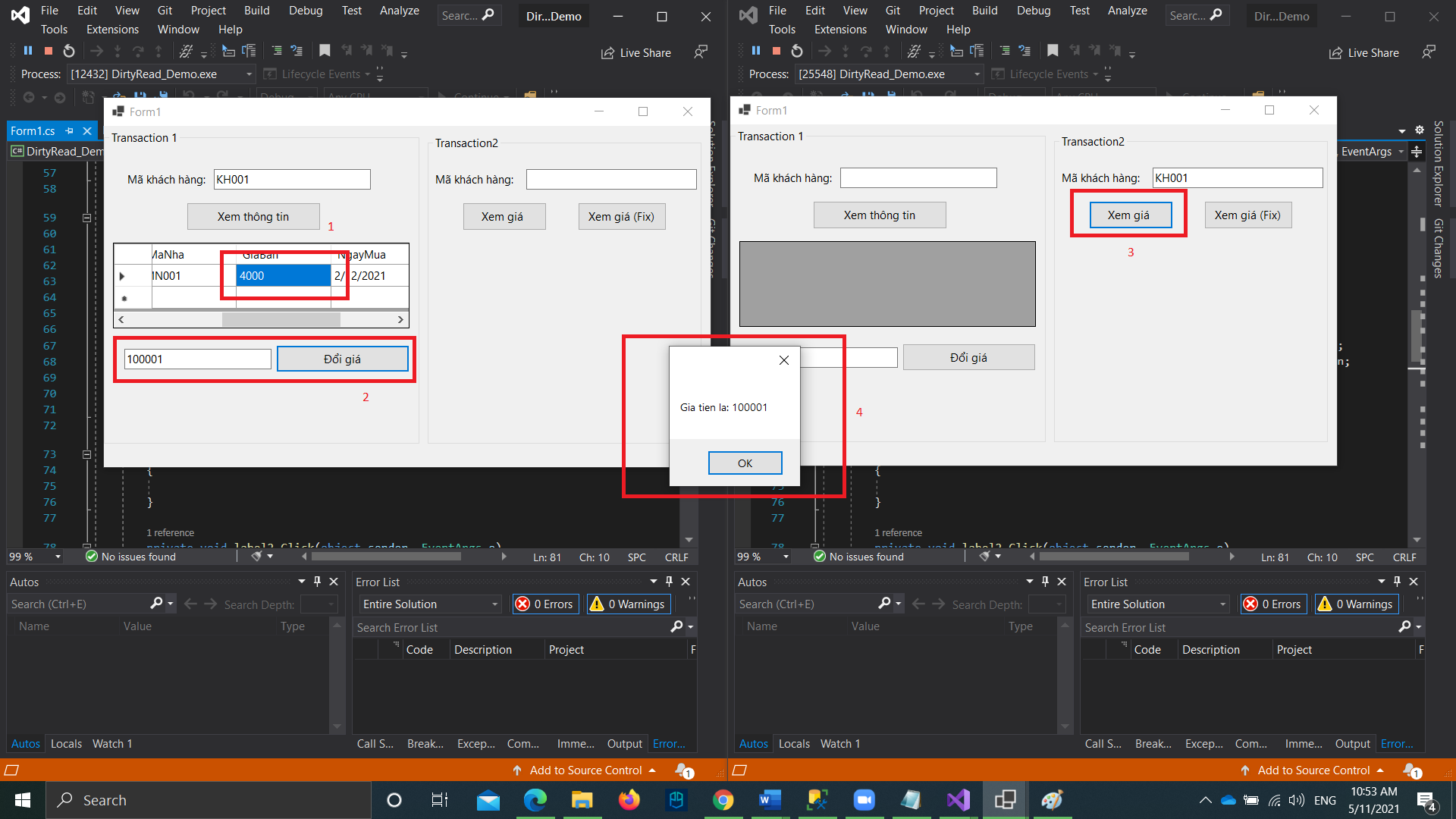
* + - * + Thiết lập mức cô lập Repeatable Read cho cả hai giao tác T để xử lý lỗi Lost Update.
      * Demo lỗi:

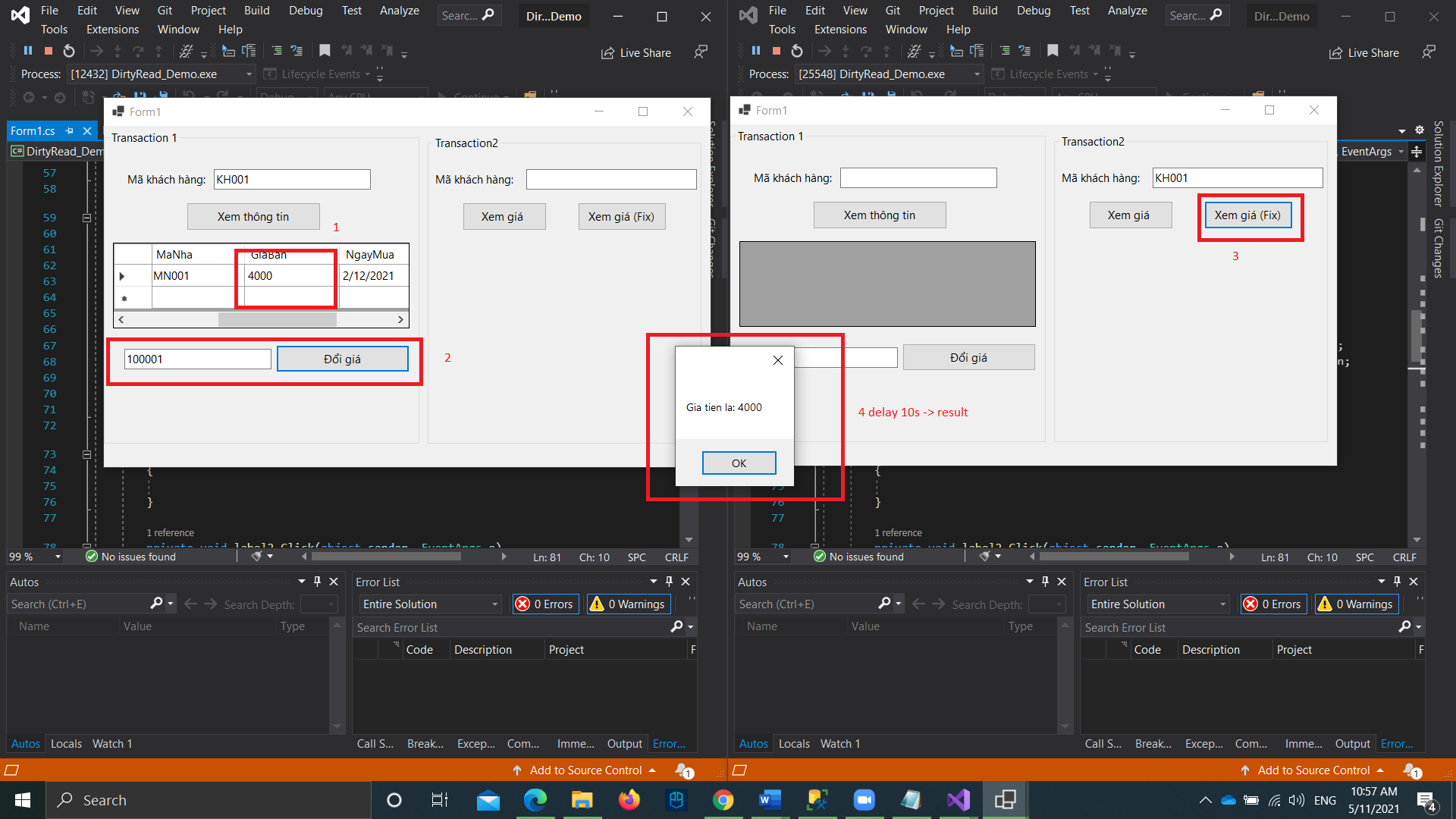


* Dirty Read: T1 với mức cô lập RC, T2 với mức cô lập URC

|  |  |  |  |
| --- | --- | --- | --- |
| T1 | Khóa | T2 | Khóa |
| Select \*  From MUA\_NHA  Where MaKH = @makh; | T1 xin khóa  S  Hệ thống  phát khóa S |  |  |
| UPDATE MUA\_NHA  Set GiaBan = @giaban Where MaKH = @makh; | T1 xin khóa X. Hệ thống  phátkhóa X |  |  |
|  |  | Declare @gianha float  Select @gianha = GiaBan From MUA\_NHA Where MaKH = @makh | T1 đang giữ khóa X  Tuy nhiên do T2 có mức  cô lập URC nên T2 không  cần xin khóa. |
| If @giaban <= 100000  Begin  Commit Transaction  End  Else  Begin  Update MUA\_NHA set GiaBan = @giaban - (GiaBan - 100000) where MaKH = @makh  Rollback Transaction  End |  | Commit |  |
| Commit |  |  |  |

* + - Thiết lập mức cô lập Read Committed cho T2 để xử lý lỗi Dirty Read.
      * Demo lỗi:

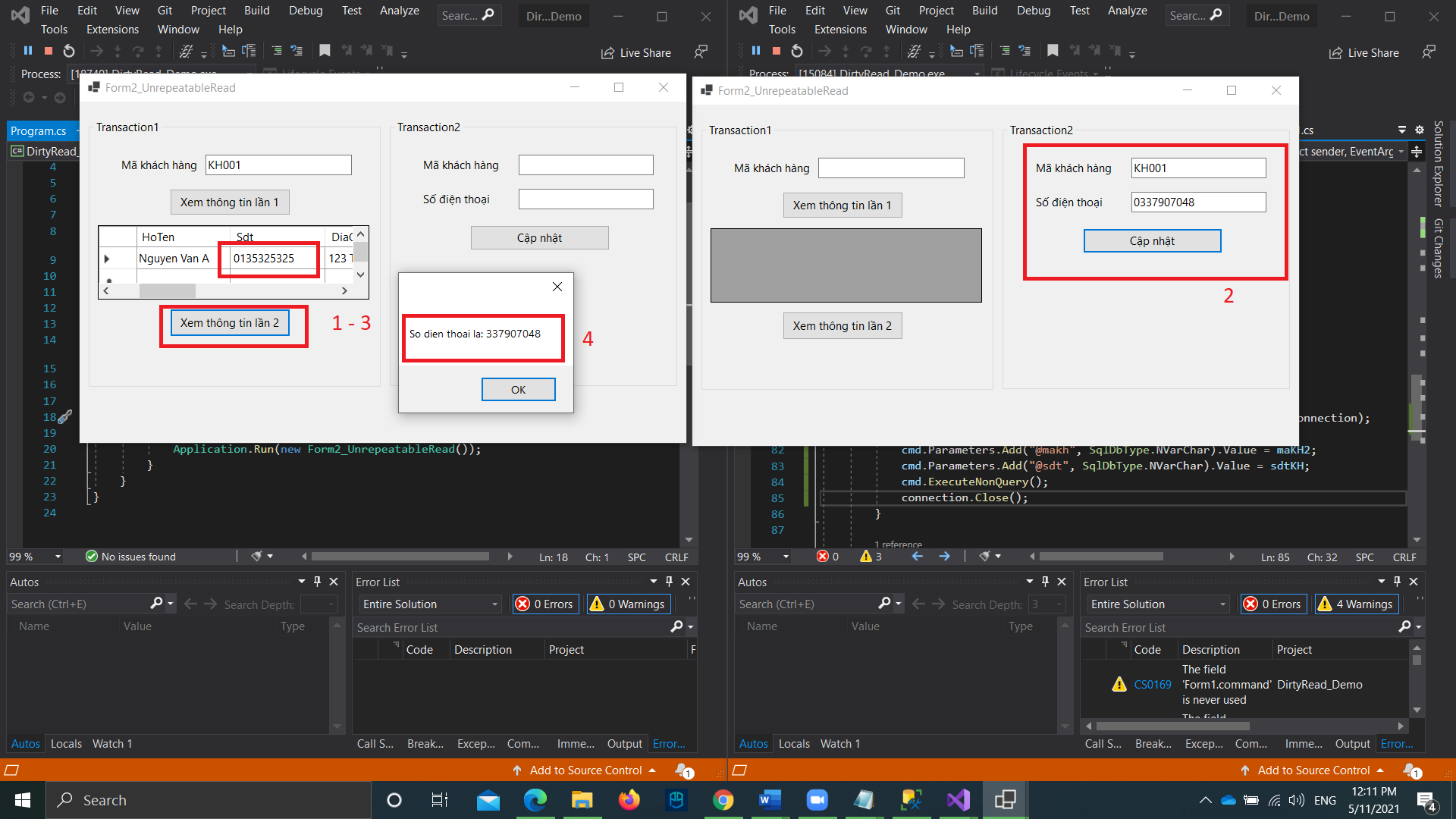


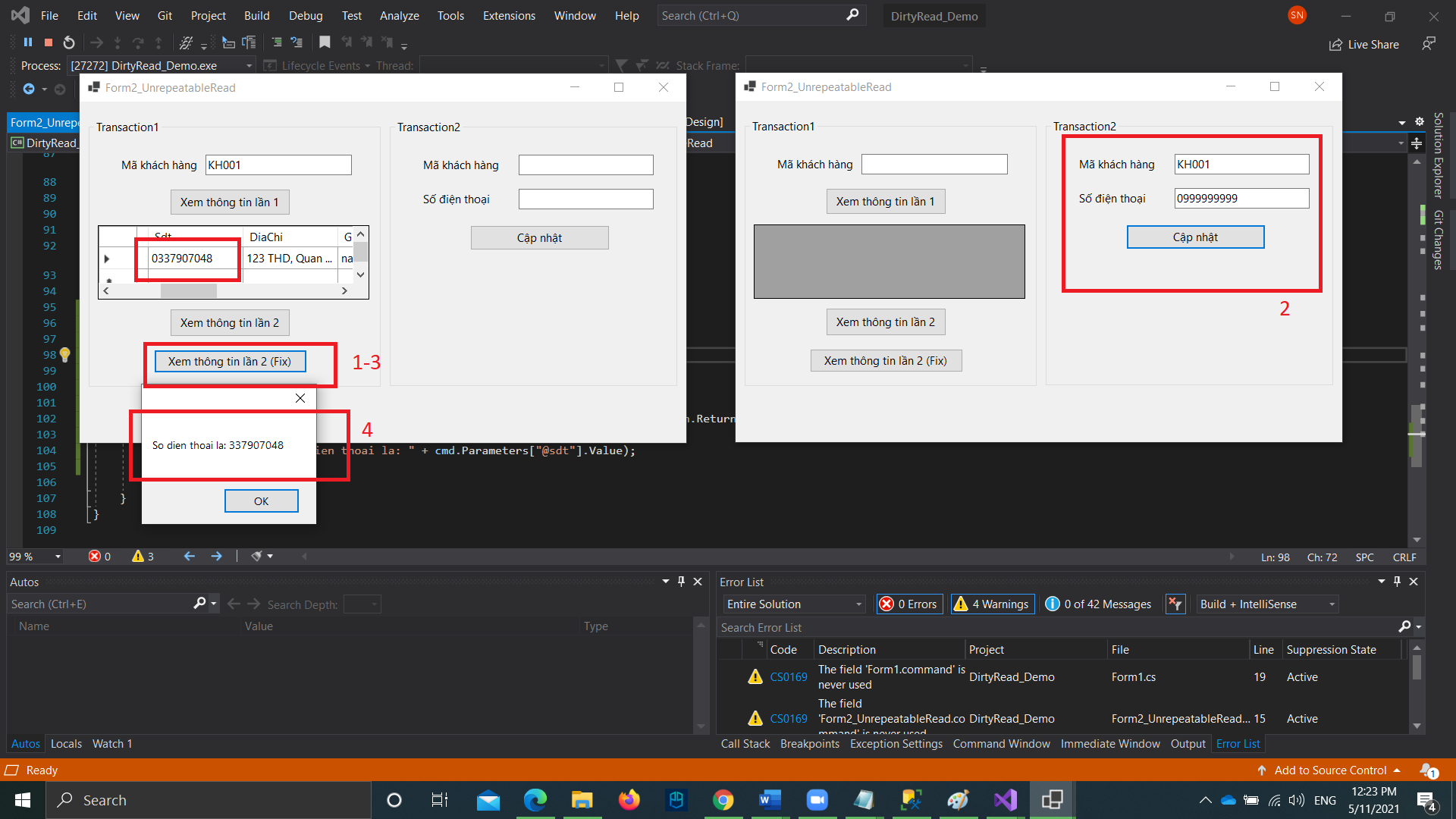


* Unrepeatables Read: T1 và T2 với mức cô lập RC

|  |  |  |  |
| --- | --- | --- | --- |
| T1 | Khóa | T2 | Khóa |
| Select \*  From KHACH\_HANG  Where MaKH = @makh; | T1 xin khóa S  Hệ thống phát  khóa S |  |  |
|  |  | UPDATE KHACH\_HANG  Set Sdt = @sdt  Where MaKH = @makh; | T2 xin khóa X  Hệ thống phát  khóa X |
| Declare @sdt nvarchar(20)  Select @sdt = Sdt  From KHACH\_HANG  Where MaKH = @makh | T1 xin khóa S  Vì T2 đang giữ  khóa X nên không  xin được khóa. Phải  đợi T2 commit mới  xin được khóa |  |  |
| Commit |  | Commit |  |

* + - Thiết lập mức cô lập Repeatable Read cho T1 để xử lý lỗi Unrepeatable Read.
      * Demo lỗi:





* Phantom Read:

|  |  |  |  |
| --- | --- | --- | --- |
| T1 | Khóa | T2 | Khóa |
| Declare @sohd int  Select @sohd = count(\*)  From HOP\_DONG  Where MaNV = @manv  Print N'So hop dong da xem lan 1: '  Print @sohd | T1 xin khóa S  Hệ thống phát khóa  S |  |  |
|  |  | Declare @ngaylap date  Set @ngaylap = N'2000-12-3'  Declare @loai nvarchar(50)  Set @loai = 'True'  Insert Into HOP\_DONG  Values(@mahd, @manv, @makh, @ngaylap, @loai, @maln, @manha) | T2 xin khóa S Sau đó xin khóa X. Hệ thống phát khóa X cho T2 |
| Select @sohd = count(\*)  From HOP\_DONG  Where MaNV = @manv  Print N'So hop dong da xem lan 2: '  Print @sohd | T1 xin khóa S, nhưng  do T2 đang giữ khóa  X nên T1 phải chờ T2 commit |  |  |
|  |  | Commit |  |
| Commit |  |  |  |

* + - Thiết lập mức cô lập Serializable cho T1 để xử lý lỗi Phantom Read.
      * Demo lỗi:

