**d**

BÁO CÁO ĐỒ ÁN THỰC HÀNH

MÔN HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU

GVHD: LƯƠNG HÁN CƠ

**MỤC LỤC:**

[THÔNG TIN NHÓM 4](#_Toc121691043)

[YÊU CẦU, PHÂN TÍCH ĐỒ ÁN 6](#_Toc121691044)

[**1.** **Yêu cầu đồ án:** 6](#_Toc121691045)

[**2.** **Phần trăm hoàn thành:** 6](#_Toc121691046)

[BẢNG THIẾT KẾ THỰC THỂ KẾT HỢP, CSDL, PHÂN QUYỀN 6](#_Toc121691047)

[**1.** **Lược đồ thực thể kết hợp ( mô hình ER):** 6](#_Toc121691048)

[**2.** **Ràng buộc dữ liệu:** 7](#_Toc121691049)

[**3.** **Mô hình dữ liệu quan hệ:** 8](#_Toc121691050)

[**4.** **Xác định [login, user], role và permission:** 8](#_Toc121691051)

[XÁC ĐỊNH CHỨC NĂNG CỦA HỆ THỐNG VÀ CÁC TÌNH HUỐNG TRANH CHẤP 9](#_Toc121691052)

[**1.** **Các chức năng của hệ thống:** 9](#_Toc121691053)

[**2.** **Thiết kế giao diện chức năng:** 12](#_Toc121691054)

[TÌNH HUỐNG TRANH CHẤP 26](#_Toc121691055)

[**1.** **Xác định các tình huống tranh chấp:** 26](#_Toc121691056)

[**2.** **Cài đặt tình huống tranh chấp** 27](#_Toc121691057)

[**a.** **Sinh viên thực hiện: Trịnh Hữu Hiệp (20120079)** 27](#_Toc121691058)

[Tình huống 1: LOST UPDATE 27](#_Toc121691059)

[Tình huống 2: UNREPEATABLE READ 33](#_Toc121691060)

[**b.** **Sinh viên thực hiện: Nguyễn Thị Hồng Nhung (20120093)** 37](#_Toc121691061)

[Tình huống 1: Lỗi Unrepeatable Read 37](#_Toc121691062)

[Tình huống 2: Lỗi Conversion Deadlock 41](#_Toc121691063)

[**c.** **Sinh viên thực hiện : Trần Thái San (20120177)** 45](#_Toc121691064)

[Tình huống 1: DIRTY READ 45](#_Toc121691065)

[Tình huống 2: PHANTOM 46](#_Toc121691066)

[**d.** **Sinh viên thực hiện: Nguyễn Nhật Trường (20120229)** 49](#_Toc121691067)

[Tình huống 1: UNREPEATABLE READ 49](#_Toc121691068)

[Tình huống 2: DIRTY READ 51](#_Toc121691069)

[Tình huống 3: CIRCLE DEADLOCK 54](#_Toc121691070)

[XỬ LÍ TÌNH HUỐNG TRANH CHẤP 57](#_Toc121691071)

[**1.** **Sinh viên thực hiện : Trịnh Hữu Hiệp (20120079)** 57](#_Toc121691072)

[Tình huống 1: LOST UPDATE 57](#_Toc121691073)

[Tình huống 2: UNREPEATABLE READ 63](#_Toc121691074)

[**2.** **Sinh viên thực hiện: Nguyễn Thị Hồng Nhung (20120093)** 66](#_Toc121691075)

[Tình huống 1: Lỗi Unrepeatable Read 66](#_Toc121691076)

[Tình huống 2: Lỗi Conversion Deadlock 70](#_Toc121691077)

[**3. Sinh viên thực hiện: Trần Thái San** 74](#_Toc121691078)

[Tình huống 1: DIRTY READ 74](#_Toc121691079)

[Tình huống 2: PHANTOM READ 75](#_Toc121691080)

[**4. Sinh viên thực hiện: Nguyễn Nhật Trường** 78](#_Toc121691081)

[Tình huống 1: UNREPEATABLE READ 78](#_Toc121691082)

[Tình huống 2: Dirty Read 80](#_Toc121691083)

[Tình huống 3: Circle Deadlock 82](#_Toc121691084)

# **THÔNG TIN NHÓM**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **MSSV** | **Họ tên** | **Công việc** | **% Hoàn thành** | **% tham gia** |
| 1 | 20120079 | Trịnh Hữu Hiệp | Thiết kế ER + RBTV+ Script cài đặt CSDL QH  Xác định chức năng và cài procedure  Xác định + mô tả lỗi Unrepeatable Read và Lost Update  Xử lý 2 tranh chấp trên  Thiết kế giao diện nhân viên, tổng hợp code | 100% | 25% |
| 2 | 20120093 | Nguyễn Thị Hồng Nhung | Thiết kế ER+ Mô hình CSDL QH  Tạo script phân quyền và cài procedure  Xác định + mô tả lỗi Unrepeatable Read và Conversion Deadlock  Xử lý 2 tranh chấp trên  Thiết kế giao diện tài xế, khách hàng | 100% | 25% |
| 3 | 20120177 | Trần Thái San | Xác định Role + tạo role  Tạo script phân quyền và cài procedure  Xác định + mô tả lỗi Dirty Read và Phantom  Xử lý 2 tranh chấp trên  Thiết kế giao diện admin, đăng nhập, đăng ký | 100% | 25% |
| 4 | 20120229 | Nguyễn Nhật Trường | Xác định + Tạo(User +Login)  Cài procedure  Xác định + mô tả lỗi Dirty Read, Unrepeatable Read, Circle Deadlock  Xử lý 3 tranh chấp trên  Thiết kế giao diện đối tác (cửa hàng) | 100% | 25% |

# YÊU CẦU, PHÂN TÍCH ĐỒ ÁN

1. **Yêu cầu đồ án:**

- Thiết kế cơ sở dữ liệu, phân tích quy trình nghiệp vụ và mô hình ER.

- Phân quyền người dùng.

- Thiết kế giao diện hệ thống, xây dựng các chức năng.

- Xác định các tình huống tranh chấp.

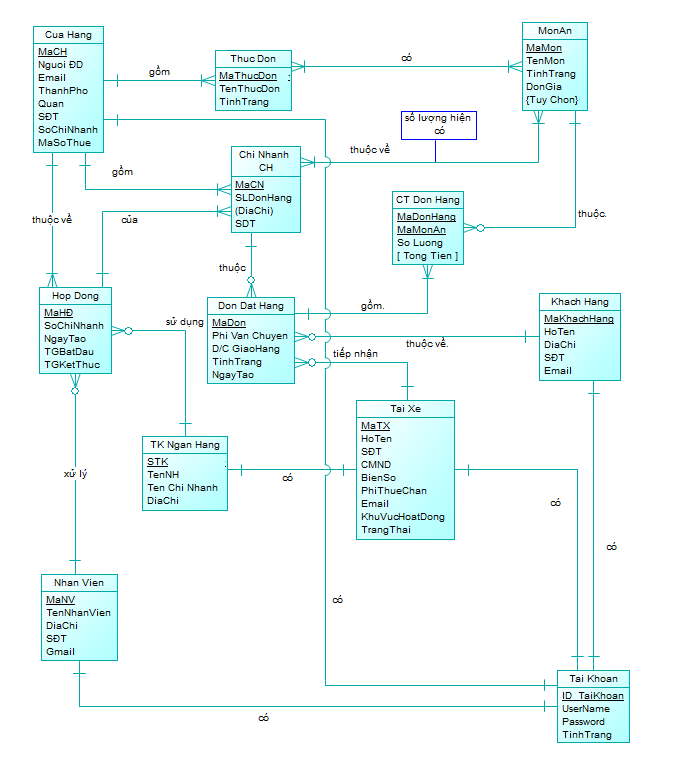
- Hướng giải quyết các tình huống tranh chấp.

1. **Phần trăm hoàn thành:**

Hoàn thành 100% yêu cầu đồ án

# BẢNG THIẾT KẾ THỰC THỂ KẾT HỢP, CSDL, PHÂN QUYỀN

1. **Lược đồ thực thể kết hợp ( mô hình ER):**



1. **Ràng buộc dữ liệu:**

<các ràng buộc toàn vẹn xác định được: miền giá trị, ràng buộc hệ thống>

Thời gian bắt đầu phải nhỏ hơn thời gian kết thúc trong hợp đồng

Trạng thái của tài xế phải là 'đã nhận đơn' hoặc null

Tình trạng của đơn hàng phải là chờ xác nhận hoặc đã xác nhận hoặc đang giao hoặc đã giao hoặc đã hủy

Số lượng món ăn trong chi tiết đơn hàng phải lớn hơn 0

Số lượng chi nhánh của cửa hàng phải lớn hơn 0

Số lượng đơn mỗi ngày trong mỗi chi nhánh phải lớn hơn 0

Tình trạng của món ăn phải là có bán hoặc hết hàng hôm nay hoặc tạm ngưng

Không thể hủy đơn hàng khi đơn hàng đang ở tình trạng là đã xác nhận hoặc đang giao hoặc đã giao

Tài xế chỉ tiếp nhận giao hàng trong khu vực đã đăng kí trước đó.

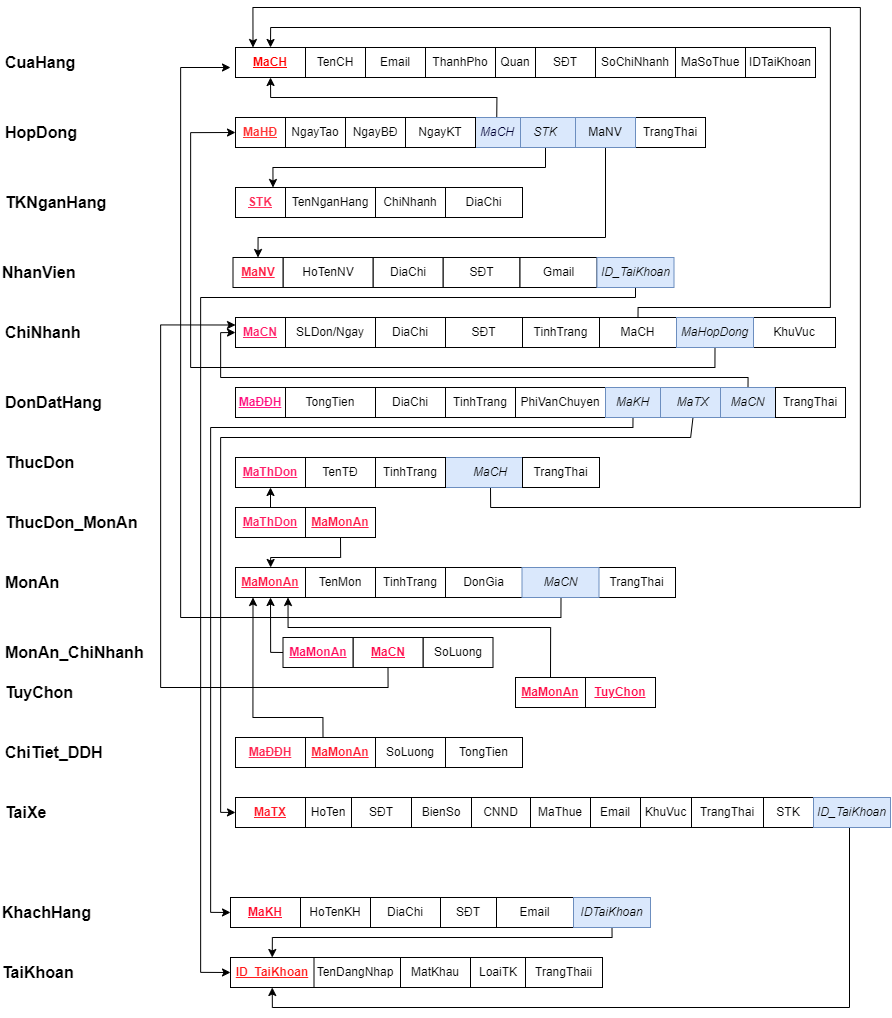
Tổng tiền trong CHITIET\_DONHANG bằng số lượng\* đơn giá của món ăn đó

Tổng tiền trong đơn đặt hàng bằng tổng tiền các chi tiết đơn hàng thuộc về về đơn đặt hàng đó + phí vận chuyển.

Tất cả các tên món ăn phải phân biệt với nhau

Trạng thái của chi nhánh cửa hàng phải là 'bình thường' hoặc 'tạm nghỉ'

1. **Mô hình dữ liệu quan hệ:**



1. **Xác định [login, user], role và permission:**
2. **Role:**

Cửa Hàng, Tài Xế, Khách Hàng, Nhân Viên, Người Quản Trị

1. **Phân quyền:**

Cửa hàng được xem, xóa, chỉnh sửa, cập nhật thông tin của cửa hàng, thực đơn, món ăn

Cửa hàng được xem đơn đặt hàng, chi tiết đơn đặt hàng, thông tin tài xế, hợp đồng, thông tin khách hàng

Cửa hàng được phép cập nhật địa chỉ của chi nhánh

Cửa hàng được phép thêm chi nhánh

Tài xế được xem, xóa, chỉnh sửa, cập nhật thông tin của tài xế

Tài xế được xem thông tin của cửa hàng, đơn đặt hàng, chi tiết đơn đặt hàng, thông tin khách hàng

Tài xế được phép cập nhập tình trạng đơn hàng của mình

Khách hàng được xem, xóa, chỉnh sửa, cập nhật thông tin của khách hàng

Khách hàng được xem thông tin chi nhánh cửa hàng, thực đơn, món ăn, đơn đặt hàng, thông tin tài xế

Nhân Viên được phép xem thông tin, cập nhật hợp đồng, xem thông tin cửa hàng

Người Quản trị được phép thêm, xóa sửa, cập nhật thông tin của toàn bộ csdl

# XÁC ĐỊNH CHỨC NĂNG CỦA HỆ THỐNG VÀ CÁC TÌNH HUỐNG TRANH CHẤP

1. **Các chức năng của hệ thống:**
2. **Chức năng dùng chung:**

|  |  |  |
| --- | --- | --- |
| **STT** | **Chức năng** | **Mô tả hoạt động** |
| ALL1 | Đăng nhập | đăng nhập vào hệ thống đặt và giao thức ăn online dựa vào tài khoản và mật khẩu đăng kí. |
| ALL2 | Đăng ký | Cung cấp thông tin, đăng ký tài khoản, sử dụng tài khoản đã tạo để đăng nhập |
| ALL3 | Nhận thông báo | Nhận thông báo khi thao tác lỗi trên hệ thống |
| ALL4 | Cập nhật tài khoản | Hệ thống hổ trợ người dùng thay đổi và cập nhật lại các thông tin tài khoản |
| ALL5 | Quên mật khẩu | Hổ trợ tạo lại mật khẩu mới trong trường hợp quên mật khẩu khi đăng nhập |
| ALL6 | Xem danh sách món ăn, cửa hàng | Hiện thị danh sách của hàng, các mon ăn, sản phẩm được đăng bán |
| ALL7 | Xem chi tiết thông tin | Hiển thi thông tin của cửa hàng, hiển thị thông tin món ăn |

1. **Phân hệ đối tác:**

|  |  |  |
| --- | --- | --- |
| **STT** | **Chức năng** | **Mô tả hoạt động** |
| AD1 | Truy cập hợp đồng | Đối tác nhận được email và mã truy cập nhận hợp đồng được cung cấp từ hệ thống sau khi đã đăng kí thành công |
| AD2 | Quản lý các chi nhánh cửa hàng | Xem thông tin các chi nhánh đã đăng kí và có quyền chỉnh sửa thông tin chi nhánh |
| AD3 | Quản lý thực đơn | tìm kiếm điều chỉnh các thực đơn bào gồm thêm, xóa, cập nhật giá bán và tình trạng món ăn |
| AD4 | Quản lý đơn đặt hàng | xem thông tin đơn hàng và cập nhật tình trạng của đơn hàng đã đặt của các chi nhánh |
| AD5 | Xem chi tiết đơn hàng | Hiển thị chi tiết các đơn hàng của các chi nhánh của đối tác |
| AD6 | Thống kê doanh thu | Hiển thị doanh thu theo tuần, tháng, năm của từng chi nhánh cửa hàng và tổng doanh thu của tất cả chi nhánh |
| AD7 | Xem xu hướng món ăn | Hiển thị món ăn xu hướng, bán chạy nhất trong tuần hoặc trong tháng |
| AD8 | Xem thông tin phản hồi | Hiện thị thông tin phản hồi của từng đơn hàng (nếu có) của khách hàng |

1. **Phân hệ khách hàng:**

|  |  |  |
| --- | --- | --- |
| **STT** | **Chức năng** | **Mô tả hoạt động** |
| CTM1 | Xem danh sách đối tác | Xem danh sách hiển thi đối tác, các chi nhánh của đối tác trong hệ thống |
| CTM2 | Xem chi tiết danh sách món | Hiện thị toàn bộ món ăn cùng đơn giá, chi tiết món ăn của chi nhánh đối tác đã được chọn |
| CTM2 | Tìm kiếm sản phẩm | tìm kiếm món ăn, đối tác bằng cách nhập tên món hoặc tên cửa hàng |
| CTM3 | Chọn món | Chọn các món và thêm vào giỏ |
| CTM4 | Quản lý giỏ hàng | Khách hàng thêm bớt các món ăn và giỏ, chọn số lượng món cùng các tùy chọn cho mỗi món |
| CTM5 | Thanh toán | Chọn hình thức thanh toán, khách hàng tạo địa chỉ nhận hàng kèm số điện thoại và chọn ‘đặt hàng’  Hệ thống hổ trợ người dùng thay đổi các thông tin về số điện thoại và địa chỉ nhận hàng |
| CTM4 | Hủy đơn hàng | Chọn hủy đơn đặt hàng khi đơn hàng chưa được xác nhận từ phía chi nhánh cửa hàng Và hủy không thàng công khi đơn hàng đã được tiếp nhận |
| CTM5 | Theo dõi đơn hàng | Theo dõi quá trình vận chuyển, tài xế giao hàng của đơn hàng đã đặt |

1. **Phân hệ tài xế:**

|  |  |  |
| --- | --- | --- |
| **STT** | **Chức năng** | **Mô tả hoạt động** |
| DRV1 | Tạo TK đăng nhập | Cung cấp thông tin, admin thực hiện tạo tài khoản |
| DRV2 | Xem danh sách đơn hàng | Hiển thị danh sách đơn hàng trong khu vực đã đăng kí của tài xế cần được giao |
| DRV3 | Xem chi tiết đơn hàng | Hiển thị thông tin đơn hàng cần được giao |
| DRV4 | Tiếp nhận và xử lý đơn hàng | Danh sách đơn hàng được hiển thị theo khu vực, tiếp nhận đơn hàng, cập nhật tình trạng ‘đang giao’ sau khi nhận để khách hàng có thể theo dõi đơn hàng đã đặt |
| DRV5 | Theo dõi thu nhập | Hiển thị danh sách các đơn hàng đã được nhận, phí vận chuyển từng đơn đã nhận |

1. **Phân hệ nhân viên:**

|  |  |  |
| --- | --- | --- |
| STT | Chức năng | Mô tả hoạt động |
| NV1 | Gia hạn hợp đồng | Thực hiện gia hạn hợp đồng, điều chỉnh ngày kêt thúc |
| NV2 | Xem danh sách hợp đồng | Hệ thống hiển thị danh sách hợp đồng đã được ký |
| NV3 | Tìm kiếm hợp đồng | Tìm kiếm hợp đồng trên hệ thống, lọc các hợp đồng theo ngày, yêu cầu gửi thông báo khi sắp hết hạn hợp đồng |
| NV4 | Duyệt hợp đồng | Hiển thị các hợp đồng mới cần ký kết , nhân viên thực hiện phê duyệt |

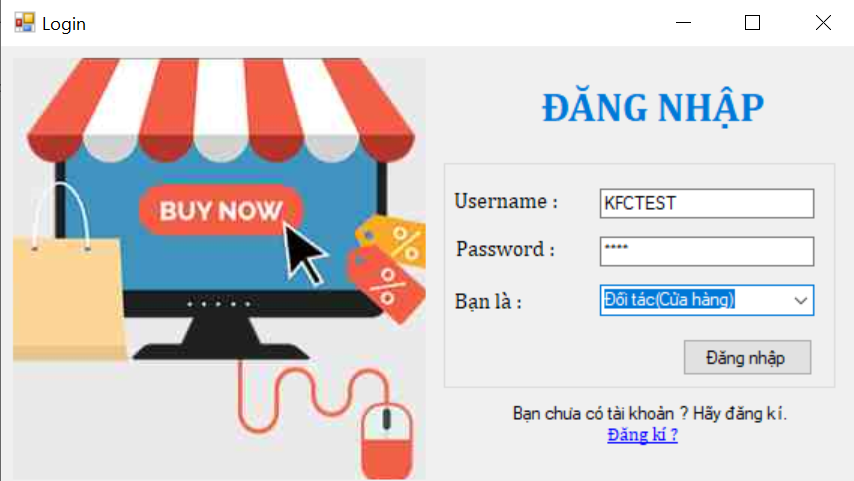
1. **Phân hệ admin:**

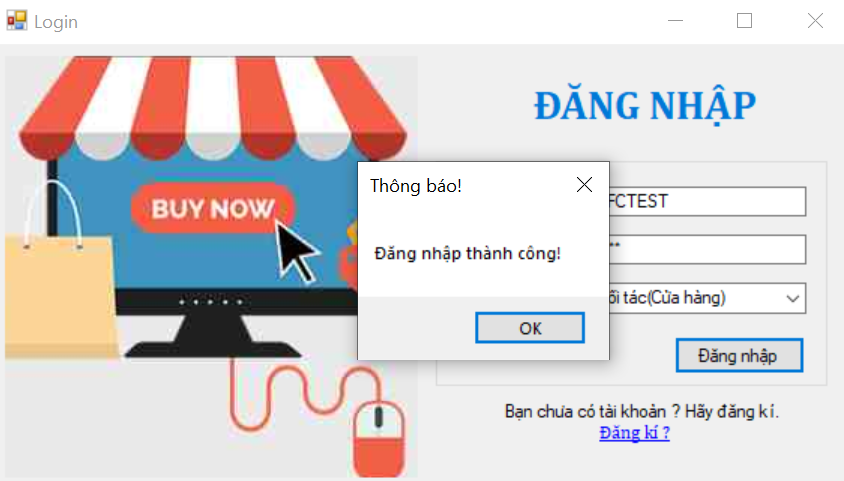
|  |  |  |
| --- | --- | --- |
| **STT** | **Chức năng** | **Mô tả hoạt động** |
| AD1 | Tạo tài khoản cho người dùng và new admin | Tạo tài khoản đăng nhập sẵn theo quy định cho người dùng và new admin |
| AD2 | Xem thông tin danh sách tài khoản | Admin xem toàn bộ danh sách các tài khoản trên hệ thống |
| AD3 | Thêm, xóa, sửa tài khoản | Admin thêm mới tài khoản, cập nhật, chỉnh sửa các tài khoản của người dùng |
| AD4 | Khóa tài khoản | Admin có quyền khóa tài khoản của người dùng và các admin khác |
| AD5 | Cấp quyền người dùng | Admin thao tac trên giao diện hệ thống, giới hạn quyền người dùng |

1. **Thiết kế giao diện chức năng:**

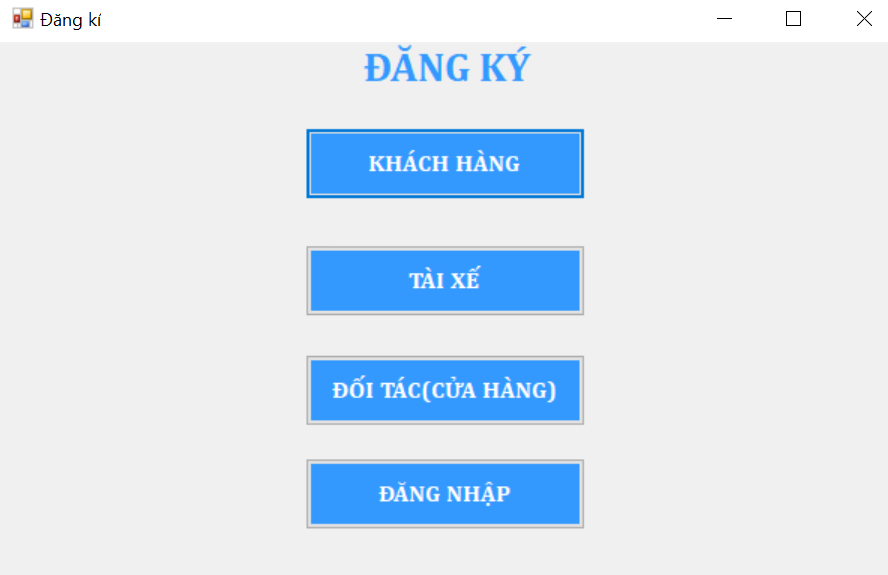
**a. Các chức năng chung:**

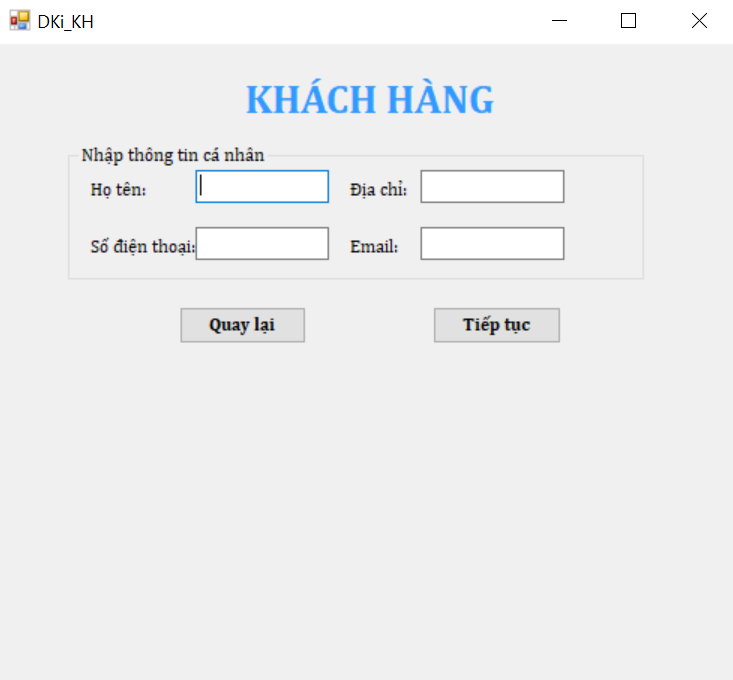
Giao diện đăng nhập:

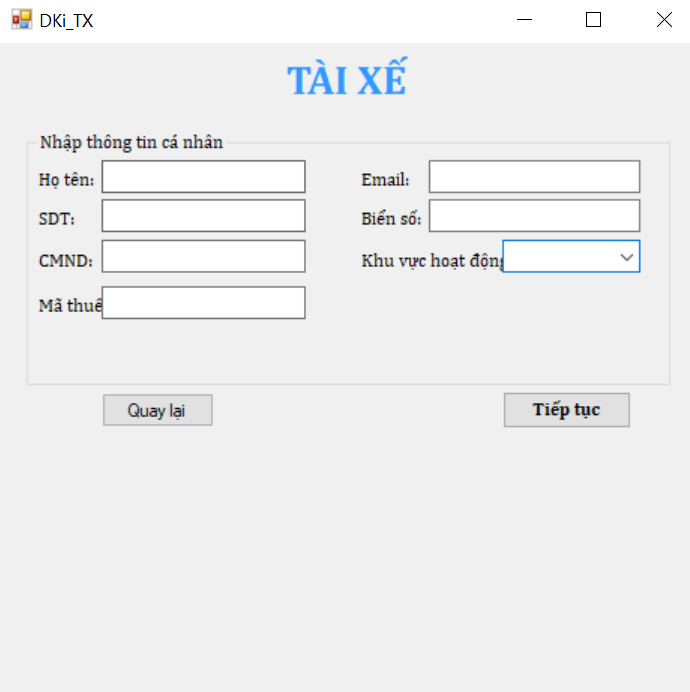


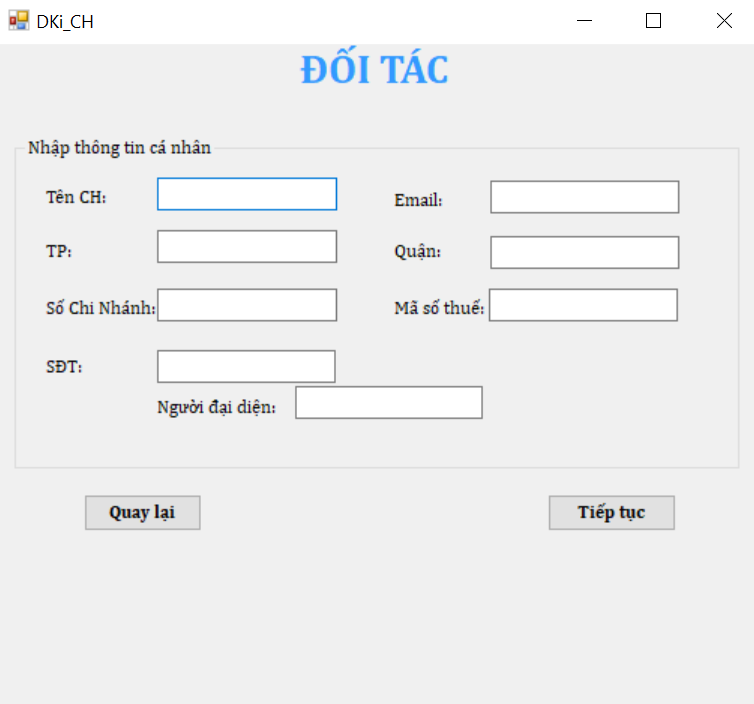


Giao diện đăng ký (dành cho đối tác (cửa hàng), khách hàng, tài xế



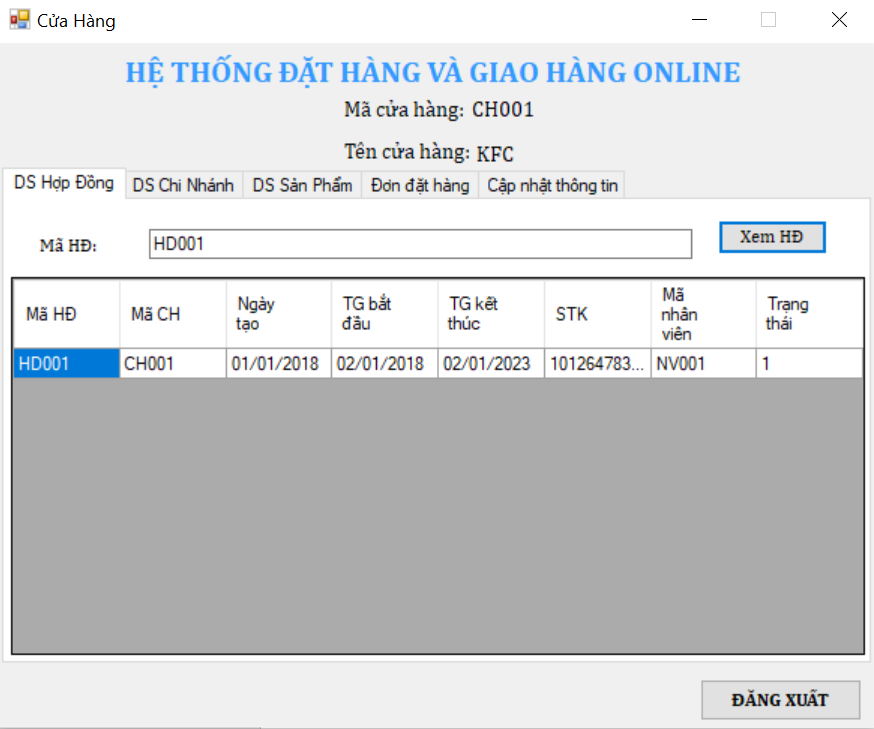




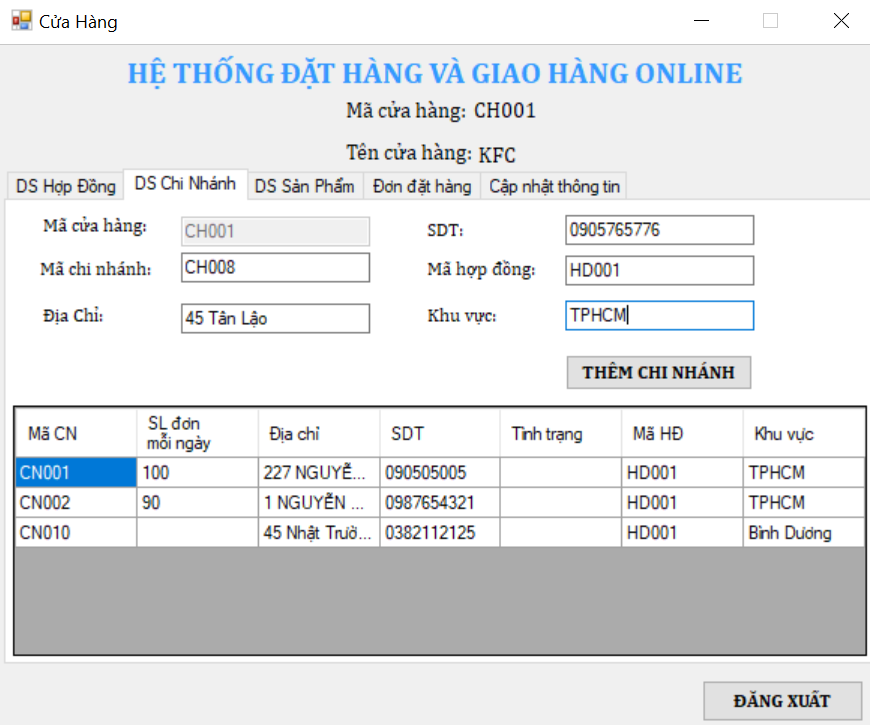


**b. Phân hệ đối tác (cửa hàng):**

Giao diện xem danh sách hợp đồng



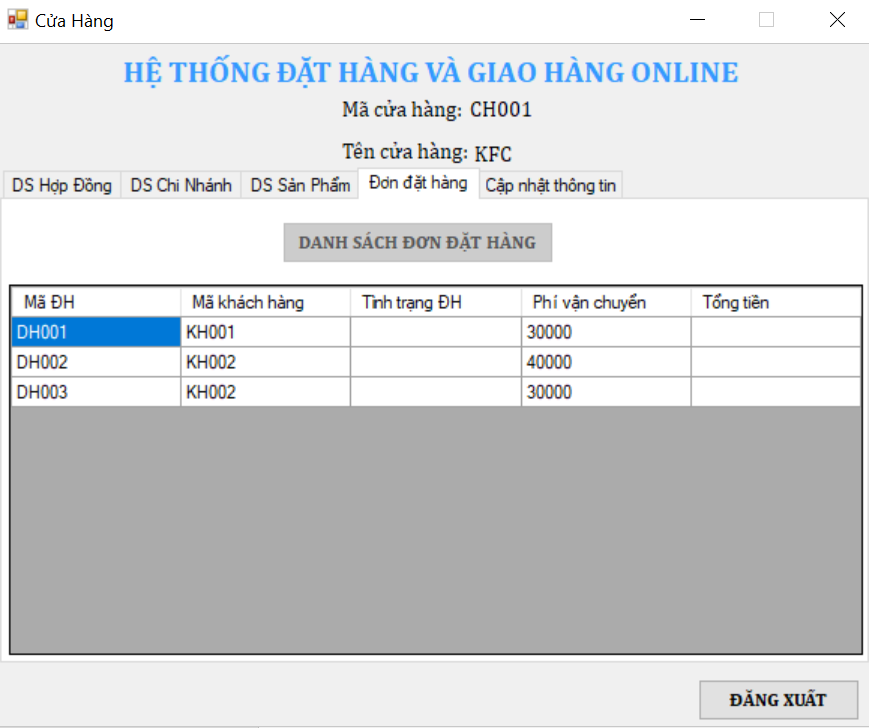
Giao diện xem danh sách chi nhánh và thêm chi nhánh mới:



Giao diện xem danh sách sản phẩm, cập nhật giá và số lượng:



Giao diện xem danh sách đơn đặt hàng:

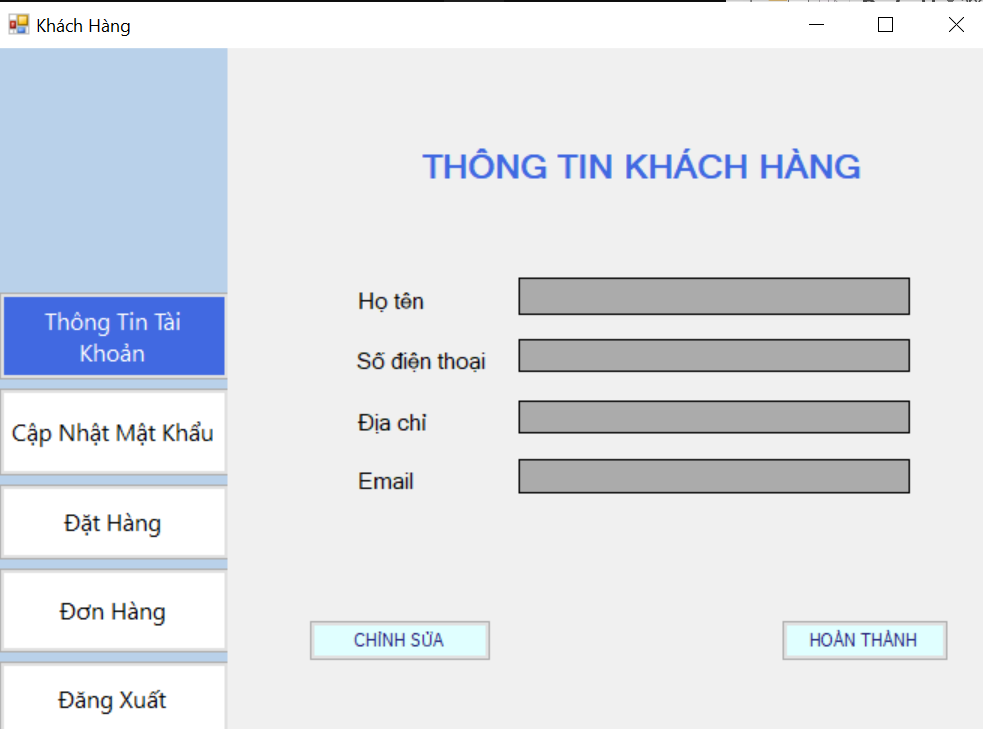


Giao diện cập nhật thông tin của cửa hàng:



**c. Phân hệ khách hàng:**

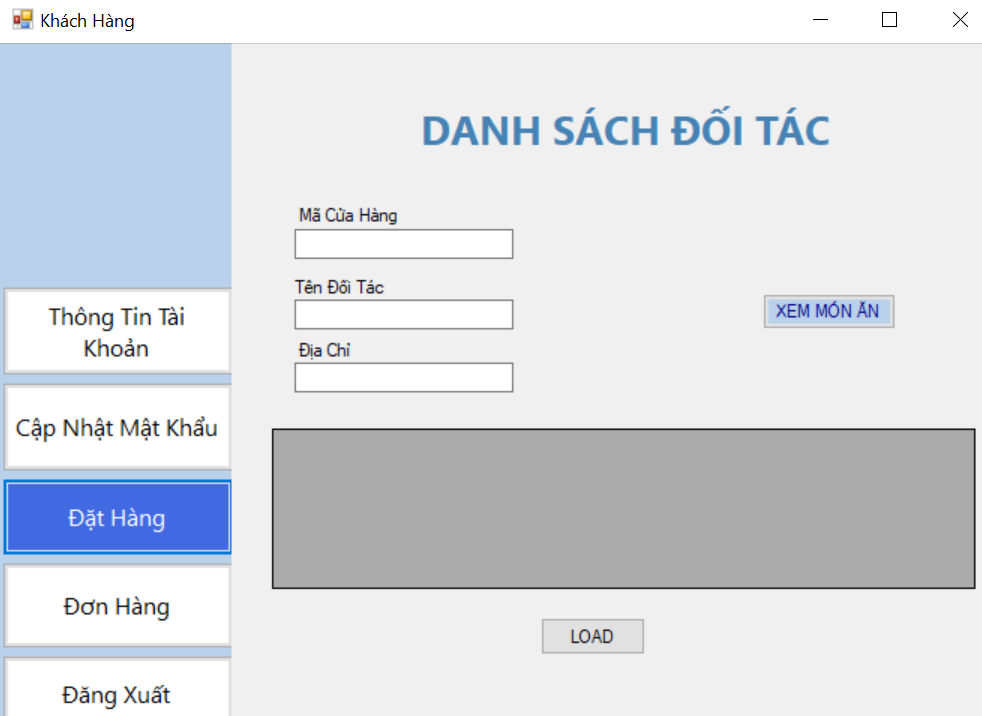
Giao diện xem thông tin và chỉnh sửa thông tin tài khoản khách hàng:



Giao diện cập nhật lại mật khẩu:



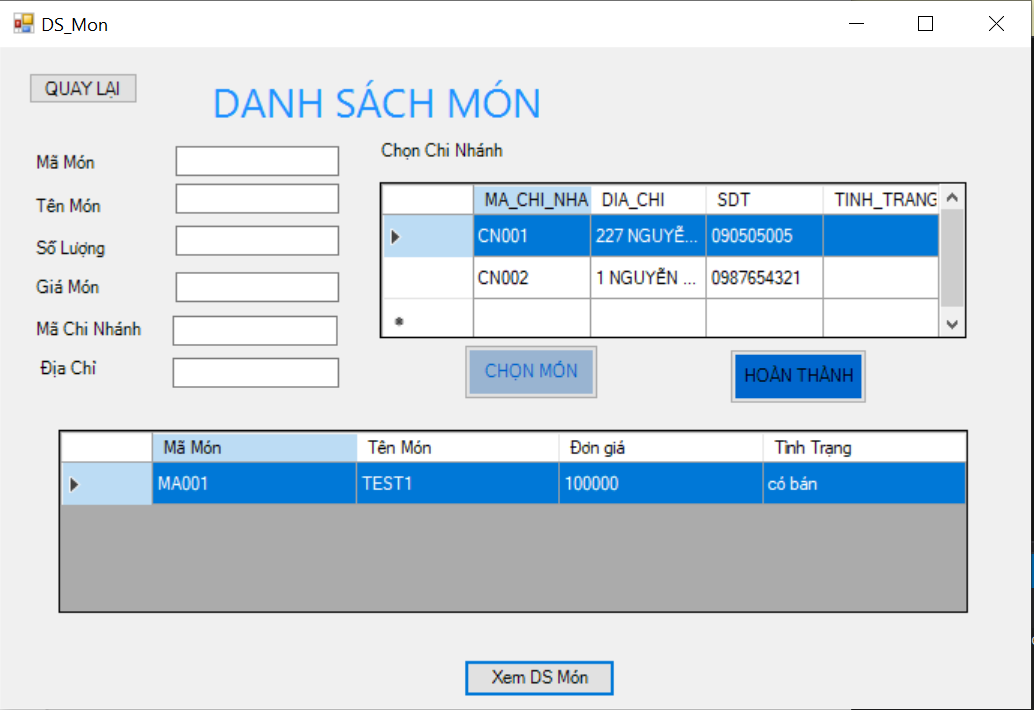
Giao diện đặt hàng:



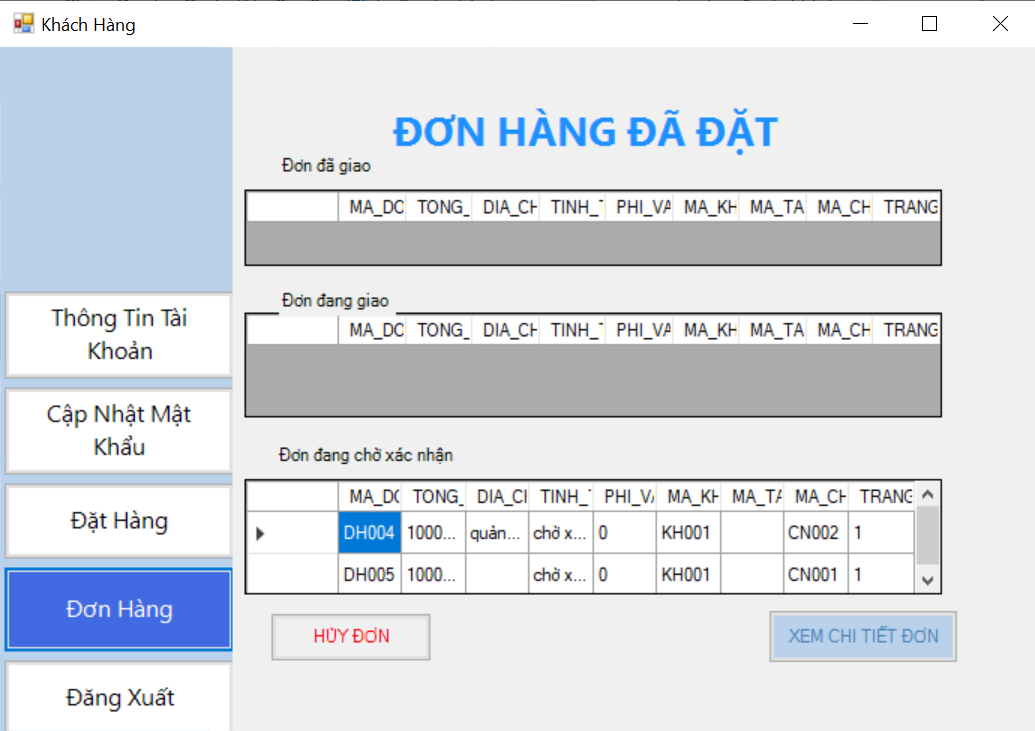
Ảnh có chứa văn bản

Mô tả được tạo tự độngGiao diện xem danh sách món:

Giao diện chọn món và đặt hàng:



Giao diện xem đơn hàng đã đặt, hủy đơn:



1. **Phân hệ tài xế**

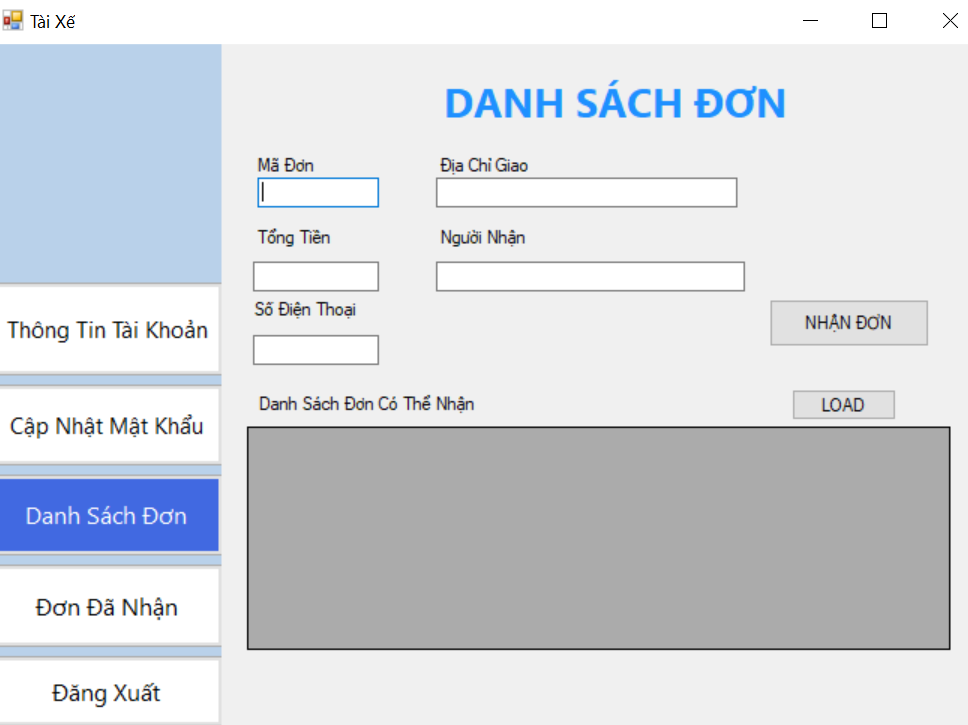
Giao diện xem thông tin tài khoản:



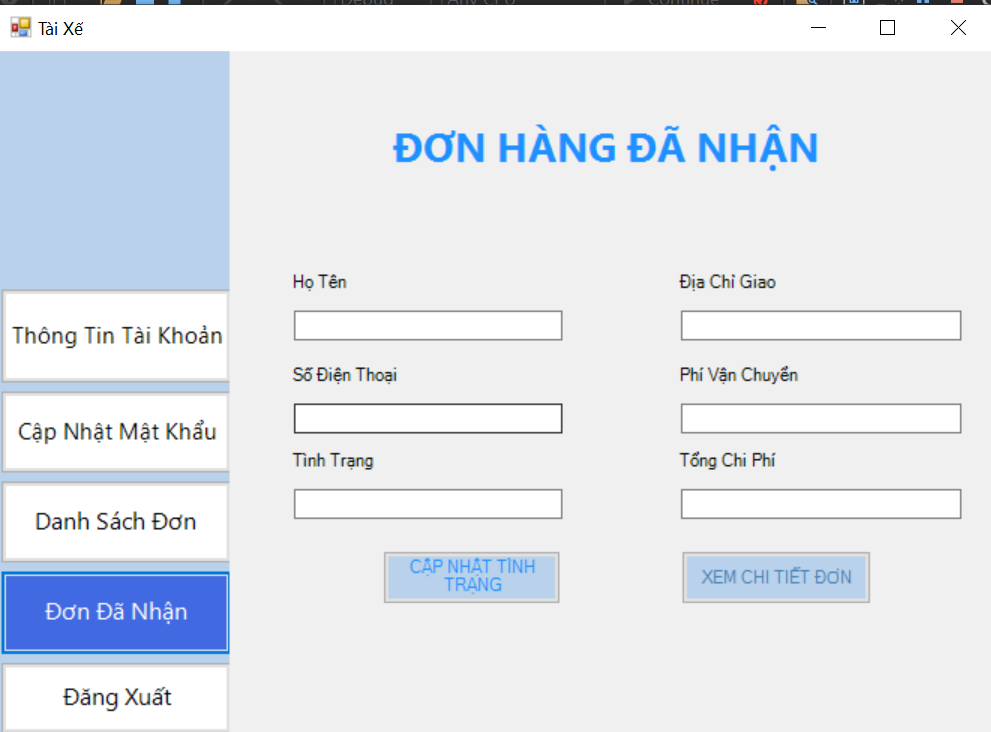
Giao diện cập nhật mật khẩu:



Giao diện xem xem sách đơn để nhận đơn hàng:

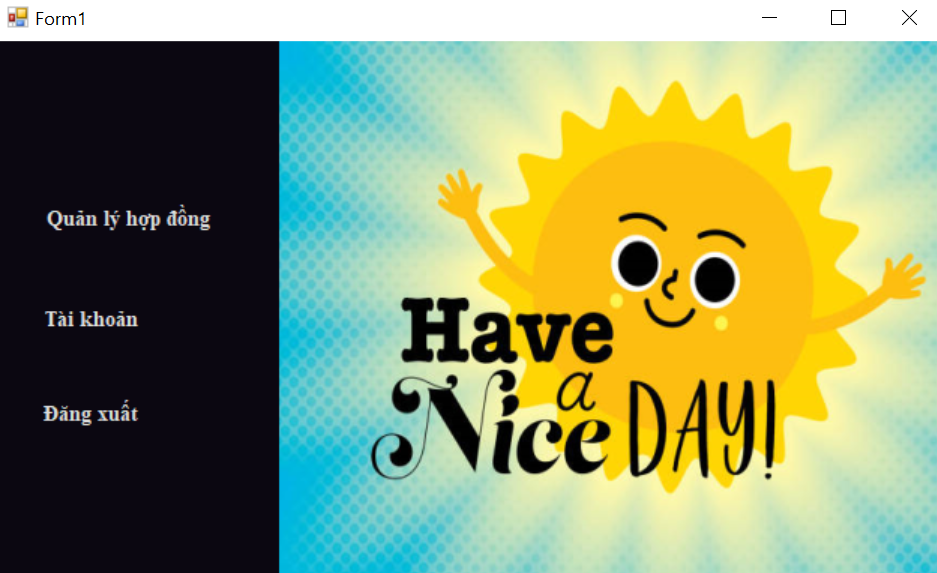


Giao diện xem đơn hàng đã đặt:

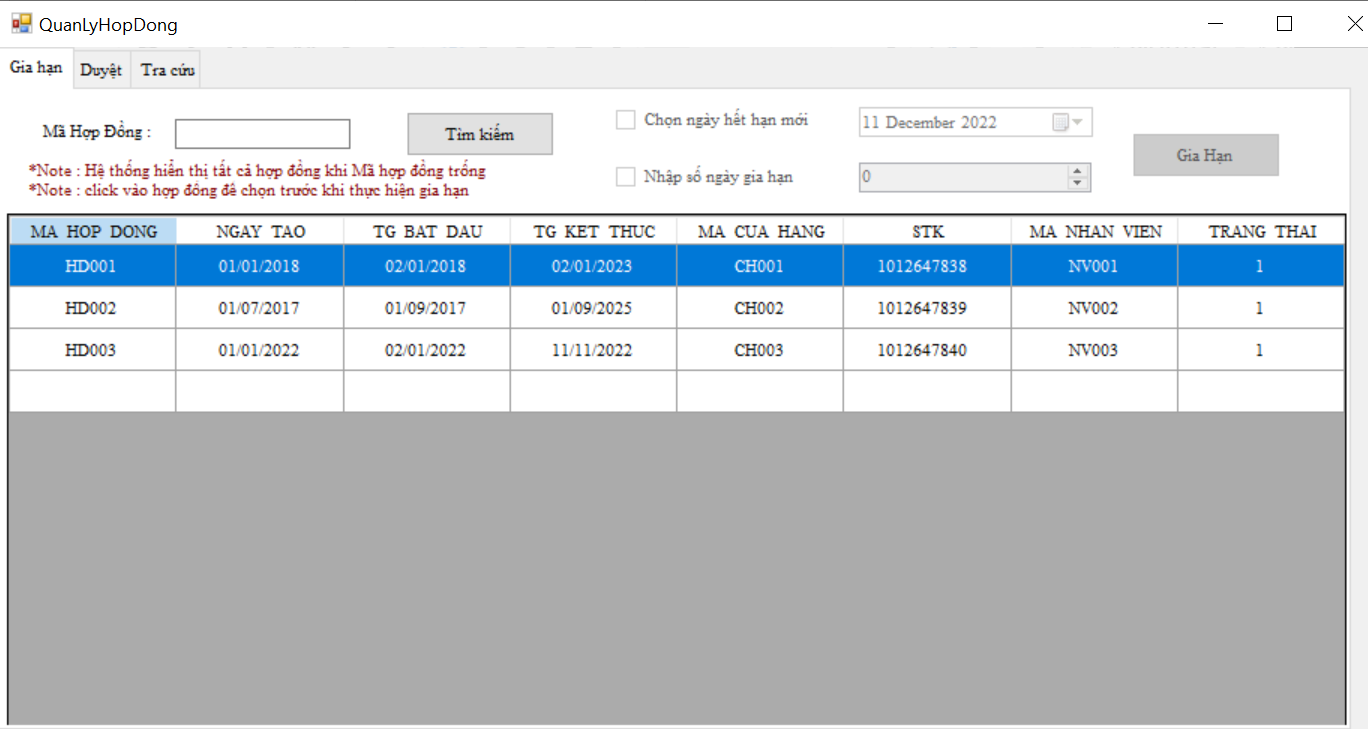


1. **Phân hệ nhân viên**

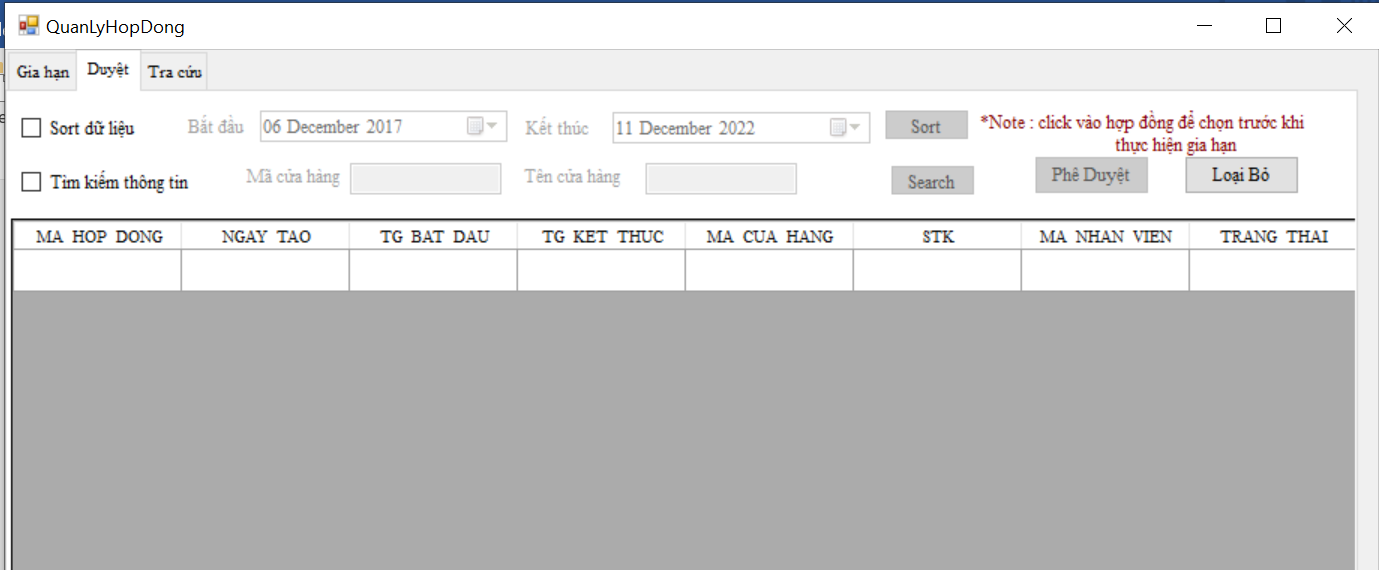
Giao diện trang chủ của nhân viên:



Giao diện quản lý hợp đồng (xem danh sách các hợp đồng, tìm kiếm hợp đồng để gia hạn):



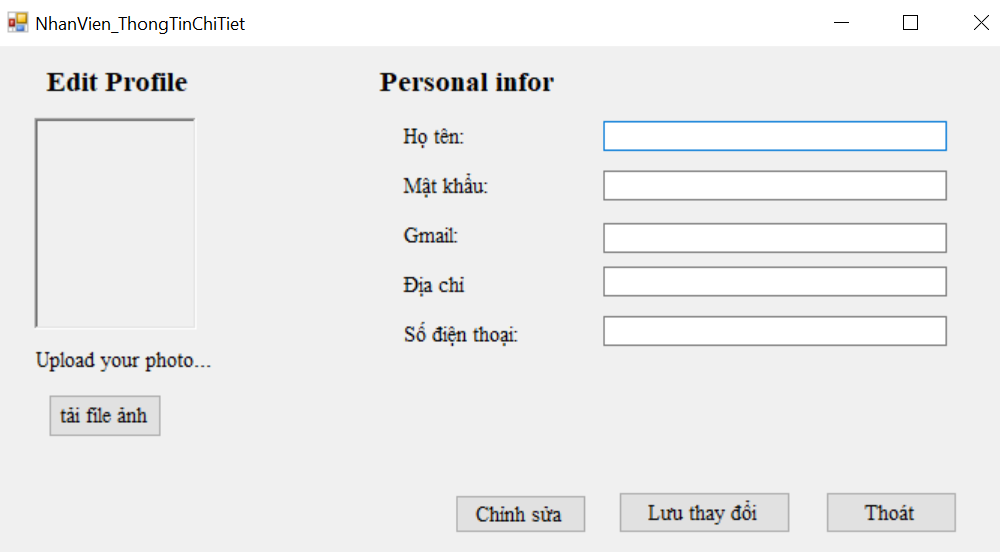
Giao diện quản lý hợp đồng (duyệt hợp đồng):



Giao diện quản lý hợp đồng (tra cứu các chi nhánh trong hợp đồng):

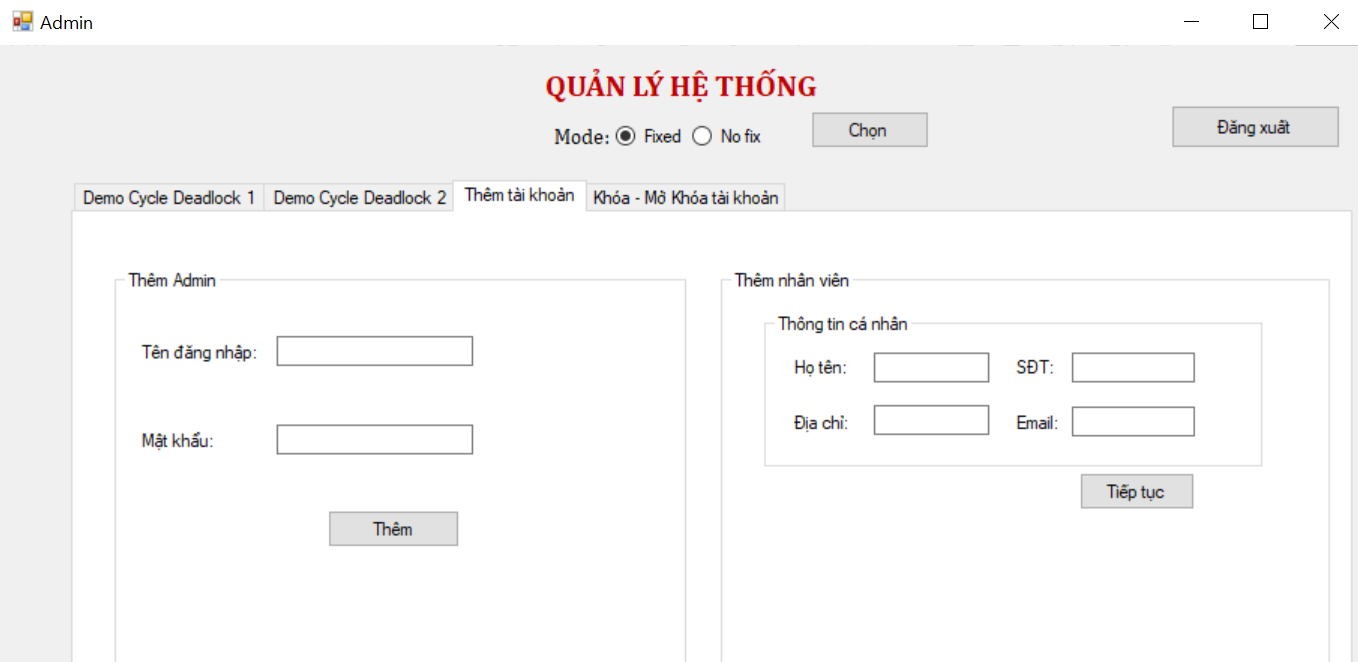


Giao diện xem thông tin cửa tài khoản nhân viên:

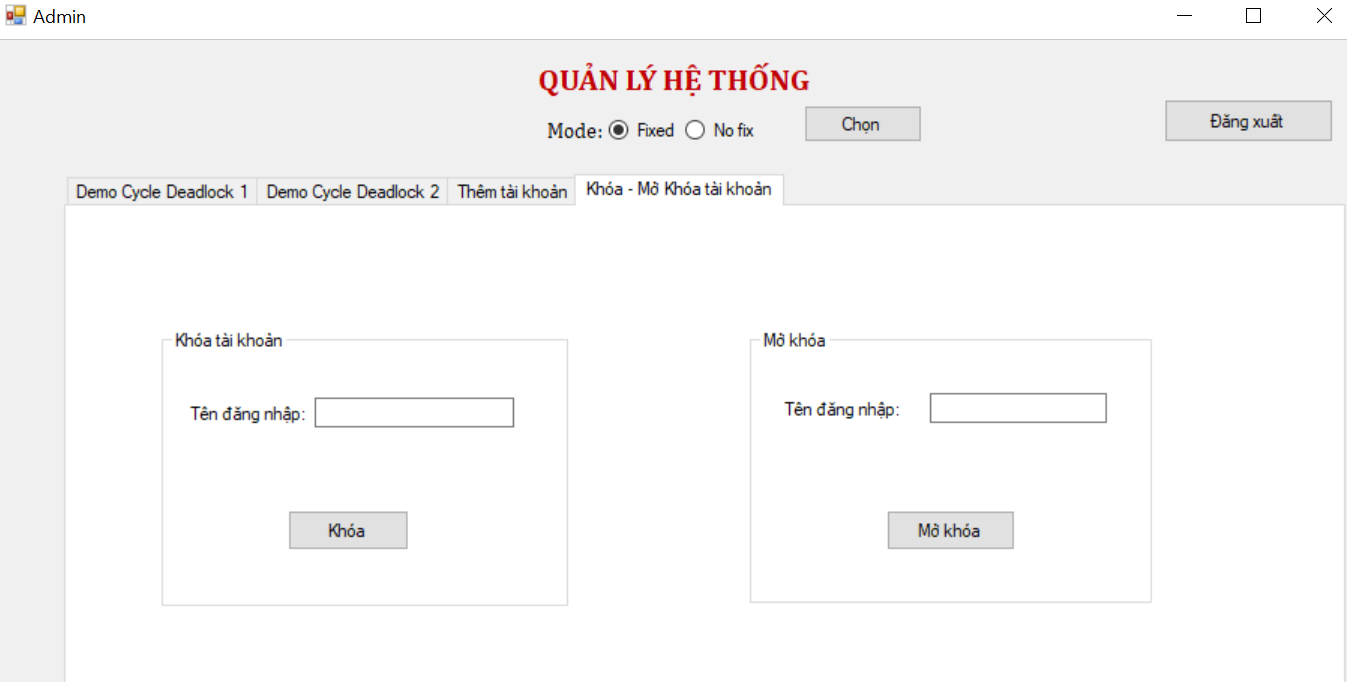


1. **Phân hệ admin**

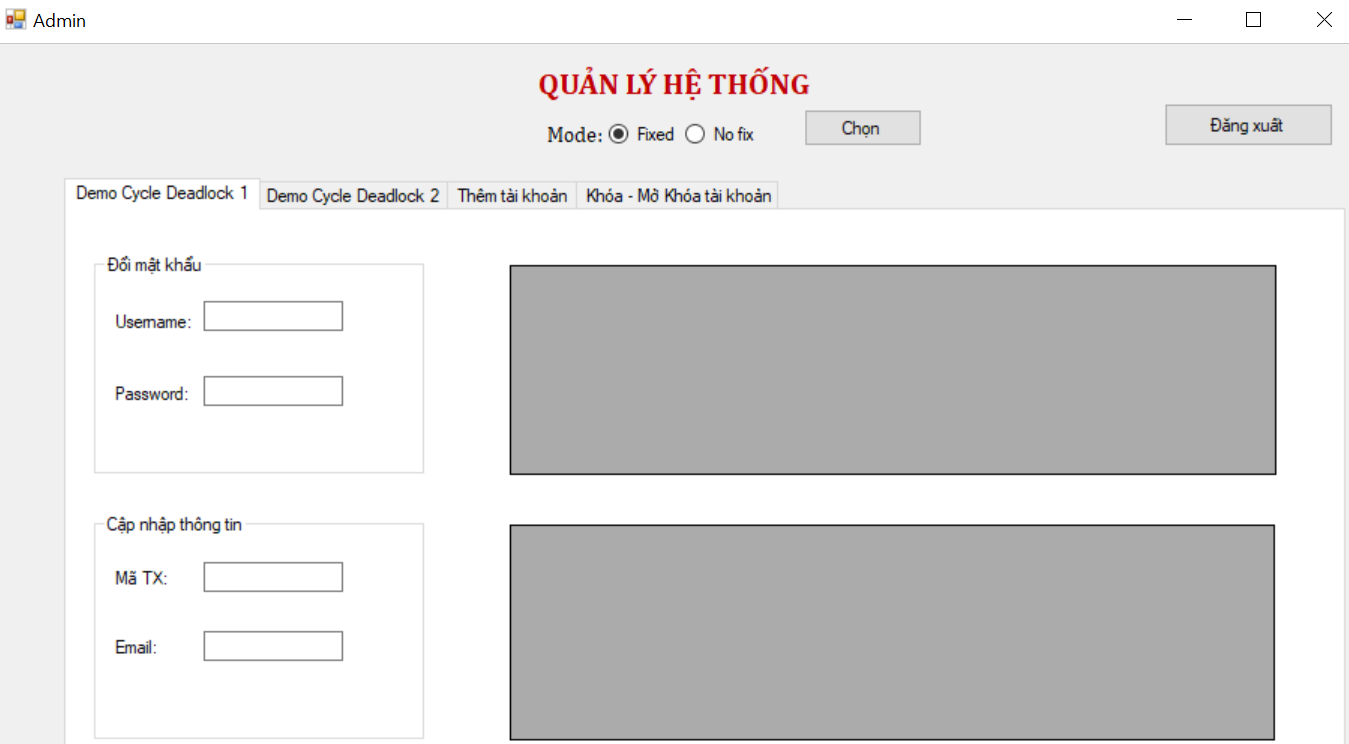
Giao diện thêm tài khoản:



Giao diện khóa, mở khóa tài khoản:



Giao diện đổi mật khẩu và cập nhật thông tin:



# 

# TÌNH HUỐNG TRANH CHẤP

1. **Xác định các tình huống tranh chấp:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **STT** | **Chức năng 1** | **Người dùng** | **Chức năng 2** | **Người dùng** | **Lỗi tranh chấp** | **Người thực hiện** |
| 1 | Cập nhật số lượng món ăn tại 1 chi nhánh | Cửa hàng (Đối tác) | Thêm đơn đặt hàng | Khách Hàng | Lost Update | Trịnh Hữu Hiệp-20120079 |
| 2 | Xem tổng tiền một món ăn | Khách hàng | Cập nhật đơn giá món ăn | Cửa hàng (đối tác) | Unrepeatable Read |
| 3 | Đăng nhập | Khách hàng | Cập nhật mật khẩu | Khách Hàng | Unrepeatable Read | Nguyễn Thị Hồng Nhung-20120093 |
| 4 | Gia hạn hợp đồng | Nhân viên | Gia hạn hợp đồng | Nhân viên | Conversion Deadlock |
| 5 | Cập nhật giá món ăn | Đối tác | Xem giá món ăn | Khách hàng | Dirty Read | Trần Thái San-20120177 |
| 6 | Thực hiên thêm 1 chi nhánh mới | Đối tác | Thực hiện xem chi nhánh của đối tác | Khách hàng | Phantom |
| 7 | Xem tình trạng đơn hàng | Khách hàng | Cập nhật tình trạng đơn hàng | Tài xế | Unrepeatable Read | Nguyễn Nhật Trường-20120229 |
| 8 | Cập nhật giá món | Đối tác | Xem giá món | Khách Hàng | Dirty Read |
| 9 | Cập nhật thông tin khách hàng (Đổi mật khẩu), cập nhât thông tin tài xế (Email) | Người quản trị 1 (admin 1) | Cập nhât thông tin tài xế (Email), cập nhật thông tin khách hàng (Đổi mật khẩu), | Người quản trị 2 (admin 2) | Circle Deadlock |

1. **Cài đặt tình huống tranh chấp**

### **Sinh viên thực hiện: Trịnh Hữu Hiệp (20120079)**

#### Tình huống 1: LOST UPDATE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ERR01: *lost update***  T1 (User = cửa hàng (đối tác)): đối tác thực hiện nhập xuất số lượng món ăn, cập nhật số lượng món ăn tại một chi nhánh  T2 (User = cửa hàng (đối tác)): đối tác thực hiện thêm mới đơn hàng (khi khách hàng mua tại chi nhánh) | | | | |
| **SP\_CH\_CAP\_NHAT\_SL\_MON\_AN** |  | **SP\_KHACH\_HANG\_DAT\_HANG\_MON\_AN** |  |
| ***Input:*** @MA\_CHI\_NHANH = ‘CN001’  @MA\_MON\_AN = ‘MA001’  @SO\_LUONG\_THEM = 100  ***Output***: 1: cập nhật thành công  0: cập nhật thất bại | ***Input***: @MA\_DON = ‘DH001’  *@MA\_MON\_AN = ‘MA001’*  *@SO\_LUONG = 10*  *@DIA\_CHI = ‘QUANG NAM’*  *@PHI\_VAN\_CHUYEN = 10000*  *@MA\_KHACH\_HANG = ‘KH001’*  *@MA\_TAI\_XE = ‘TX001’*  *@MA\_CHI\_NHANH = ‘CN001’*  ***Output***: 1: cập nhật thành công  0: cập nhật thất bại |
| SET TRANSACTION ISOLATION  LEVEL **READ COMMITTED** | SET TRANSACTION ISOLATION  LEVEL **READ COMMITTED** |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra thông tin tồn tại của @MA\_CHI\_NHANH và @MA\_MON\_AN  IF NOT EXISTS  (SELECT \* FROM MON\_AN\_CHI\_NHANH WHERE MA\_CHI\_NHANH = @MA\_CHI\_NHANH AND MA\_MON\_AN = @MA\_MON\_AN) --(1)  BEGIN  PRINT @MA\_CHI\_NHANH +', '+  @MA\_MON\_AN + N' KHÔNG HỢP LỆ !'  ROLLBACK TRAN  RETURN 0  END | **S(MON\_AN\_CHI\_**  **NHANH)**  //Xin khoá đọc trên bảng MON\_AN\_  CHI\_NHAnH  // (xin thành công)  Sau khi đọc xong thực hiện giải phóng khóa  Do mức cô lập (RC) |  |  |
| B2: Kiểm tra @SO\_LUONG\_THEM  IF @SO\_LUONG\_THEM <1  BEGIN  PRINT N'SỐ LƯỢNG THÊM KHÔNG HỢP LỆ !'  ROLLBACK TRAN  RETURN 0  END | //kiểm tra @SO\_LUONG\_THEM hợp lệ |  |  |
| DECLARE @SO\_LUONG\_HIEN\_CO INT  SET @SO\_LUONG\_HIEN\_CO =  (SELECT SELECT SO\_LUONG FROM MON\_AN\_CHI\_NHANH WHERE MA\_CHI\_NHANH = @MA\_CHI\_NHANH AND MA\_MON\_AN = @MA\_MON\_AN) BEGIN  PRINT ' SỐ LƯỢNG HIỆN CÓ KHÔNG HỢP LỆ !'  ROLLBACK TRAN  RETURN 0  END | **S(MON\_AN\_CHI**  **\_NHANH)**  //Xin khoá đọc trên bảng MON\_AN\_  CHI\_NHAnH  // (xin thành công)  Sau khi đọc xong thực hiện giải phóng khóa |  |  | |
| WAITFOR DELAY '00:00:10' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1.1: Kiểm tra thông tin tồn tại của @MA\_KHACH\_HANG ,@MA\_TAI\_XE, @MA\_CHI\_NHANH  IF( @MA\_KHACH\_HANG NOT IN( SELECT MA\_KHACH\_HANG FROM KHACHHANG) OR  @MA\_TAI\_XE NOT IN( SELECT MA\_TAI\_XE FROM TAIXE) OR @MA\_CHI\_NHANH NOT IN  ( SELECT MA\_CHI\_NHANH FROM CHINHANH))  BEGIN  PRINT N'KHÔNG TỒN TẠI ' + @MA\_KHACH\_HANG +N' HOẶC '+ @MA\_TAI\_XE +N' HOẶC '+ @MA\_CHI\_NHANH  ROLLBACK TRAN  RETURN 0  END | **S(KHACHHANG)**  **S(TAIXE)**  **S(CHINHANH)**  //Xin khoá đọc trên bảng KHACHHANG,TAIXE,  CHINHANH  (xin thành công) Sau khi đọc xong thực hiện giải phóng khóa  Do mức cô lập (RC) |
|  |  | B1.2: Kiểm tra thông tin tồn tại của @MA\_DON và @MA\_MON\_AN  IF(EXISTS( SELECT \* FROM DONDATHANG DH WHERE DH.MA\_DON = @MA\_DON))  BEGIN  PRINT @MA\_DON + N' ĐÃ TỒN TẠI'  ROLLBACK TRAN  RETURN 0  END  IF(@MA\_MON\_AN NOT IN  (SELECT MA\_MON\_AN FROM MON\_AN WHERE TINH\_TRANG = N'có bán'))  BEGIN  PRINT @MA\_MON\_AN + N' KHÔNG TỒN TẠI'  ROLLBACK TRAN  RETURN 0  END | **S(DON\_DAT\_HANG)**  //Xin khoá đọc trên bảng (xin thành công) DON\_DAT\_HANG  đọc xong thực hiện giải phóng khóa(RC)  **S(MON\_AN)**  //Xin khoá đọc trên bảng MON\_AN  (xin thành công) đọc xong thực hiện giải phóng khóa(RC) |
|  |  | B2: Kiểm tra thông tin tồn tại @SO\_lUONG  IF (@SO\_LUONG > ( SELECT A.SO\_LUONG FROM MON\_AN\_CHI\_NHANH A WHERE A.MA\_CHI\_NHANH = @MA\_CHI\_NHANH AND A.MA\_MON\_AN =@MA\_MON\_AN)) --(2)  BEGIN  PRINT N'SỐ LƯỢNG HIỆN TẠI CỦA CHI NHÁNH '+ @MA\_CHI\_NHANH+' KHÔNG ĐỦ ĐÁP ỨNG'  ROLLBACK TRAN  RETURN 0  END | **S(MON\_AN\_CHI\_NHANH)**  //Xin khoá đọc trên bảng MON\_AN\_CHI  \_NHANH  (xin thành công)  đọc xong thực hiện giải phóng khóa |
|  |  | B3: lấy đơn giá món ăn , tạo biến tổng tiền trước khi insert  DECLARE @DON\_GIA\_MON FLOAT  SET @DON\_GIA\_MON = (SELECT DON\_GIA FROM MON\_AN WHERE MA\_MON\_AN = @MA\_MON\_AN)  DECLARE @TONG\_TIEN FLOAT  SET @TONG\_TIEN = @DON\_GIA\_MON \* @SO\_LUONG | **S(MON\_AN)**  //Xin khoá đọc trên bảng MON\_AN  (xin thành công)  đọc xong thực hiện giải phóng khóa  ( mức cô lập RC ) |
|  |  | B4: insert dữ liệu vào bảng DONDATHANG VÀ CHITIET\_DONHANH  -- INSERT DON DAT HÀNG  INSERT INTO DONDATHANG VALUES(@MA\_DON,@TONG\_TIEN,@DIA\_CHI,N'chờ xác nhận', @PHI\_VAN\_CHUYEN, @MA\_KHACH\_HANG,@MA\_TAI\_XE,@MA\_CHI\_NHANH,1)  -- INSERT CHI TIẾT ĐƠN HÀNG  INSERT INTO CHITIET\_DONHANG VALUES(@MA\_DON , @MA\_MON\_AN, @SO\_LUONG, @TONG\_TIEN) | **X(DONDATHANG)**  //Xin khoá ghi trên bảng DONDATHANG  (xin thành công)  Và giữ khóa đến hết giao tác do mức cô lập RC  **X(CHITIET\_**  **DON\_HANG)**  //Xin khoá ghi trên bảng CHITIET\_DONHANG  (xin thành công)  Và giữ khóa đến hết giao tác do mức cô lập RC |
|  |  | B5: cập nhật số lượng món ăn tại chi nhánh khách hàng mua hàng (‘CN001)  UPDATE MON\_AN\_CHI\_NHANH  SET SO\_LUONG = SO\_LUONG - @SO\_LUONG --(3)  WHERE @MA\_CHI\_NHANH = MA\_CHI\_NHANH AND @MA\_MON\_AN = MA\_MON\_AN | **X(MON\_AN**  **\_CHI\_NHANH)**  //Xin khoá GHI trên bảng MON\_AN\_  CHI\_NHANH  (xin thành công)  Và giữ khóa đến hết giao tác do mức cô lập RC |
|  |  | COMMIT TRAN  PRINT N'CẬP NHẬT THÀNH CÔNG'  RETURN 1 |  |
| B3:cập nhật số lượng món ăn tại chi nhánh  UPDATE MON\_AN\_CHI\_NHANH  SET SO\_LUONG = @SO\_LUONG\_HIEN\_CO + @SO\_LUONG\_THEM  WHERE MA\_CHI\_NHANH = @MA\_CHI\_NHANH AND MA\_MON\_AN = @MA\_MON\_AN | **X(MON\_AN**  **\_CHI\_NHANH)**  //Xin khoá ghi trên bản MON\_AN\_CHI  \_NHANH  (xin thành công)  Giữ đến hết giao tác  Do T2 đã commit và nhả khóa X |  |  |
| COMMIT TRAN  PRINT N'CẬP NHẬT THÀNH CÔNG'  RETURN 1 |  |  |  |

#### Tình huống 2: UNREPEATABLE READ

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: UNREPEATABLE READ**  T1 (User = cửa hàng(đối tác)): đối tác thực hiện cập nhật đơn giá của món ăn (@MA\_MON\_AN = ‘MA001’)  T2 (User = khách hàng): xem tổng tiền của một món ăn (@MA\_MON\_AN = ‘MA001’) với số lượng nhất định | | | |
| **SP\_KH\_XEM\_TONG\_TIEN\_MOT\_MON\_AN** | **Khóa** | **SP\_CH\_CAP\_NHAT\_DON\_GIA\_MOT\_MON\_AN** | **Khóa** |
| ***Input:*** @MA\_CUA\_HANG = ‘CH001’  @MA\_MON\_AN = ‘MA001’  @SOUONG = 10  ***Output***: 1: hiển thị thành công  0: thất bại | ***Input***: @MA\_CUA\_HANG = ‘CH001’  @MA\_MON\_AN = ‘MA001’  *@NEW\_DONGIA = 100000*  ***Output***: 1: cập nhật thành công  0: cập nhật thất bại |
| SET TRANSACTION ISOLATION  LEVEL **READ COMMITTED** | SET TRANSACTION ISOLATION  LEVEL **READ COMMITTED** |
| BEGIN TRAN |  |  |  |
| B1: kiểm tra @MA\_CUA\_HANG tồn tại  IF(@MA\_CUA\_HANG NOT IN ( SELECT MA\_CUA\_HANG FROM CUAHANG ))  BEGIN  PRINT @MA\_CUA\_HANG + N' KHÔNG HỢP LỆ !'  ROLLBACK TRAN  RETURN 0  END | **S(CUAHANG)**  //Xin khoá đọc trên bảng  (xin thành công) đọc xong thực hiện giải phóng khóa  (mức cô lập RC ) |  |  |
| B1: kiểm tra @MA\_MON\_AN tồn tại và thuộc về của hàng ‘CH001’  IF(NOT EXISTS ( SELECT \* FROM MON\_AN MA WHERE MA.MA\_MON\_AN = @MA\_MON\_AN AND MA.TINH\_TRANG = N'có bán' AND MA.TRANG\_THAI = 1 AND MA.MA\_CUA\_HANG = @MA\_CUA\_HANG))  BEGIN  PRINT @MA\_MON\_AN + N' KHÔNG HỢP LỆ !'  ROLLBACK TRAN  RETURN 0  END | **S(MON\_AN)**  //Xin khoá đọc trên bảng MON\_AN  (xin thành công)  đọc xong thực hiện giải phóng khóa  ( mức cô lập RC ) |  |  |
| WAITFOR DELAY '00:00:15' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | IF(@MA\_CUA\_HANG NOT IN ( SELECT MA\_CUA\_HANG FROM CUAHANG))  BEGIN  PRINT @MA\_CUA\_HANG + N' KHÔNG HỢP LỆ !'  ROLLBACK TRAN  RETURN 0  END | **S(CUAHANG)**  // Xin khoá đọc trên bảng CUAHANG  (xin thành công)  đọc xong thực hiện giải phóng khóa  ( do mức cô lập RC ) |
|  |  | IF(NOT EXISTS ( SELECT \* FROM MON\_AN MA WHERE MA.MA\_MON\_AN = @MA\_MON\_AN AND TRANG\_THAI !=0 AND MA.MA\_CUA\_HANG = @MA\_CUA\_HANG))  BEGIN  PRINT @MA\_MON\_AN + N' KHÔNG HỢP LỆ !'  ROLLBACK TRAN  RETURN 0  END --(1) | **S(MON\_AN)**  // Xin khoá đọc trên bảng MON\_AN  (xin thành công)  đọc xong thực hiện giải phóng khóa  ( do mức cô lập RC ) |
|  |  | UPDATE MON\_AN  SET DON\_GIA = @NEW\_DONGIA  WHERE MA\_MON\_AN = @MA\_MON\_AN1 | **X(MON\_AN)**  //Xin khoá ghi trên bảng MON\_AN  (xin thành công)  Giữ khóa đến hết giao tác |
|  |  | COMMIT TRAN  PRINT N'CẬP NHẬT THÀNH CÔNG'  RETURN 1 |  |
| SELECT @SO\_LUONG\* DON\_GIA  FROM MON\_AN  WHERE @MA\_MON\_AN = MA\_MON\_AN | **X(MON\_AN)**  //Xin khoá ghi trên bảng MON\_AN (xin thành công do T2 đã commit và nhả khóa ghi trên bảng MON\_AN)  Giữ khóa đến hết giao tác |  |  |
| COMMIT TRAN  PRINT N'HIỂN THỊ THÀNH CÔNG'  RETURN 1 |  |  |  |

### **Sinh viên thực hiện: Nguyễn Thị Hồng Nhung (20120093)**

#### Tình huống 1: Lỗi Unrepeatable Read

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ERR01: UNREPEATABLE READ**  T1 (User = Khách hàng): khách hàng A thực hiện đăng nhập  T2 (User = Khách hàng): ở thiết bị khác của khách hàng A lại cập nhật mật khẩu | | | | |
| **dang\_nhap** | **Khóa** | **cap\_nhat\_mat\_khau** | **Khóa** | |
| ***Input: tên đăng nhập (@ten\_tk)***  ***Mật khẩu (@mat\_khau)***  ***Output***: nếu thành công: output ra table gồm:  loại tài khoản, mã khách hàng | ***Input***: ***tên đăng nhập (@ten\_tk)***  ***Mật khẩu (@mat\_khau)***  ***Mật khẩu mới (@mat\_khau\_moi)***  ***Output***: nếu thành công mật khẩu mới được cập nhật, nếu không thành công rollback |
| SET TRANSACTION ISOLATION  LEVEL **READ COMMITTED** | SET TRANSACTION ISOLATION  LEVEL **READ COMMITTED** |
| BEGIN TRAN |  |  |  | |
| B1: Kiểm tra tên đăng nhập có tồn tại không  if not exists (select tk.TEN\_DANG\_NHAP from TAIKHOAN tk where tk.TEN\_DANG\_NHAP=@ten\_tk)  begin  print N'Không tồn tại tên đăng nhập'  rollback tran  return -1  end | **S(TAIKHOAN)**  //Xin khóa đọc trên bảng **TAIKHOAN**  sau khi đọc xong thì giải phóng khóa do mức cô lập RC  (Xin thành công) |  |  | |
| B2: Kiểm tra mật khẩu có đúng không  if not exists (select tk.TEN\_DANG\_NHAP from TAIKHOAN tk where tk.TEN\_DANG\_NHAP=@ten\_tk and tk.MAT\_KHAU= @mat\_khau)  begin  print N'Sai mật khẩu'  rollback tran  return -1  end | **S(TAIKHOAN)**  //Xin khóa đọc trên bảng **TAIKHOAN**  sau khi đọc xong thì giải phóng khóa do mức cô lập RC  (Xin thành công) |  |  | |
| declare @loai\_tk int  set @loai\_tk = NULL  SET @loai\_tk = (SELECT TOP 1 tk.LOAI\_TK  FROM TAIKHOAN tk  WHERE tk.TEN\_DANG\_NHAP = @ten\_tk AND tk.MAT\_KHAU = @mat\_khau AND tk.TRANG\_THAI = 1) | **S(TAIKHOAN)**  //Xin khóa đọc trên bảng **TAIKHOAN**  sau khi đọc xong thì giải phóng khóa do mức cô lập RC  (Xin thành công) |  |  |
| WAITFOR DELAY '00:00:10' |  |  |  | |
|  |  | BEGIN TRAN |  | |
|  |  | B1: Kiểm tra thông tin (1) ……  if not exists (select tk.TEN\_DANG\_NHAP from TAIKHOAN tk where tk.TEN\_DANG\_NHAP=@ten\_tk)  begin  print N'Sai tên đăng nhập'  rollback tran  return -1  end | **S(TAIKHOAN)**  **/**/Xin khóa đọc trên bảng **TAIKHOAN**  (Xin thành công vì ở mức cô lập RC thì T1 đọc xong sẽ giải phóng khóa S nên T2 có thể xin được khóa S) | |
|  |  | B2: Thống kê Chuyến Xe  if not exists ( select tk.TEN\_DANG\_NHAP from TAIKHOAN tk where tk.TEN\_DANG\_NHAP=@ten\_tk and tk.MAT\_KHAU= @mat\_khau)  begin  print N'Sai mật khẩu. Vui lòng nhập đúng mật khẩu để sửa đổi thông tin'  rollback tran  return -1  end | **S(TAIKHOAN)**  **/**/Xin khóa đọc trên bảng **TAIKHOAN**  (Xin thành công ) như trên | |
|  |  | update TAIKHOAN  set MAT\_KHAU=@mat\_khau\_moi  where TEN\_DANG\_NHAP= @ten\_tk  print N'Đổi mật khẩu thành công'  end try  begin catch  print N'Thay đổi mật khẩu thất bại'  rollback tran  end catch | **X(TAIKHOAN)**  //Xin khóa ghi trên bảng TAIKHOAN(xin thành công cho T2 đến khi hoàn thàng T2) | |
|  |  | COMMIT TRAN |  | |
| IF @loai\_tk IS NOT NULL  begin  if @loai\_tk=0  begin  SELECT distinct @loai\_tk AS 'loai\_tk', CH.MA\_CUA\_HANG AS 'ma'  FROM CUAHANG CH  WHERE CH.MA\_CUA\_HANG = @ten\_tk  COMMIT TRAN  RETURN  end  else if @loai\_tk=1  BEGIN  SELECT distinct @loai\_tk AS 'loai\_tk', KH.MA\_KHACH\_HANG AS 'ma'  FROM KHACHHANG KH, TAIKHOAN tk  WHERE KH.TEN\_DANG\_NHAP = @ten\_tk and tk.MAT\_KHAU=@mat\_khau  COMMIT TRAN  RETURN  END  else if @loai\_tk=2  BEGIN  SELECT distinct @loai\_tk AS 'loai\_tk', TX.MA\_TAI\_XE AS 'ma'  FROM TAIXE TX, TAIKHOAN tk  WHERE TX.TEN\_DANG\_NHAP = @ten\_tk and tk.MAT\_KHAU=@mat\_khau  COMMIT TRAN  RETURN  END  else if @loai\_tk=3  begin  select distinct @loai\_tk as 'loai\_tk', NV.MA\_NHAN\_VIEN from NHANVIEN NV, TAIKHOAN tk  where NV.TEN\_DANG\_NHAP=@ten\_tk and tk.MAT\_KHAU=@mat\_khau  end  else if @loai\_tk=3  begin  declare @ad char(10)  set @ad= 'admin'  select distinct @loai\_tk as 'loai\_tk', tk.TEN\_DANG\_NHAP from TAIKHOAN tk  where tk.TEN\_DANG\_NHAP=@ten\_tk and tk.MAT\_KHAU=@mat\_khau  end  end | **S(TAIKHOAN)**  //Xin khóa đọc trên bảng **TAIKHOAN**  sau khi đọc xong thì giải phóng khóa do mức cô lập RC  và T2 đã COMMIT xong  (Xin thành công) |  |  |
| end try  begin catch  print N'Đăng nhập thất bại'  rollback tran  end catch  commit tran  go |  |  |  |
| COMMIT TRAN |  |  |  |

#### Tình huống 2: Lỗi Conversion Deadlock

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ERR02: CONVERSION DEADLOCK**  T1 (User = Nhân viên): nhân viên A thực hiện gia hạn hợp đồng của cửa hàng C  T2 (User = Nhân viên): nhân viên B cũng thực hiện gia hạn hợp đồng của cửa hàng C | | | | |
| **Gia\_han\_hop\_dong** | **Khóa** | **Gia\_han\_hop\_dong** | **Khóa** |
| ***Input:*** Mã hợp đồng A (*@maHD*), số lượng ngày thêm gia hạn hợp đồng (*@so\_ngay\_them)*  ***Output***: thời gian kết thúc được gia hạn thêm với số ngày gia hạn hợp đồng thêm | ***Input:*** Mã hợp đồng A(*@maHD*), số lượng ngày thêm gia hạn hợp đồng (*@so\_ngay\_them)*  ***Output***: thời gian kết thúc được gia hạn thêm với số ngày gia hạn hợp đồng thêm |
| SET TRANSACTION ISOLATION LEVEL **SERIALIZABLE** | SET TRANSACTION ISOLATION LEVEL **SERIALIZABLE** |  |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra mã Hợp đồng có tồn tại không  if @maHD not in (select hd.MA\_HOP\_DONG  from HOPDONG hd  where hd.MA\_HOP\_DONG=@maHD)  begin  print N'Không tồn tại hợp đồng này'  rollback tran  end | **S(HOPDONG)**  //Xin khóa đọc bảng HOPDONG  (Xin thành công) |  |  |
| B2: tạo biến @ngay\_hien\_tai và set bằng ngày cuối hợp đồng  declare @ngay\_hien\_tai date  set @ngay\_hien\_tai=( select top 1 hd.TG\_KET\_THUC  from HOPDONG hd  where hd.MA\_HOP\_DONG=@maHD) | **S(HOPDONG)**  //Xin khóa đọc bảng HOPDONG  (Xin thành công) |  |  |
| if @ngay\_hien\_tai is not null  begin  -- bắt đầu chờ  WAITFOR DELAY '00:00:15' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1.1: Kiểm tra mã Hợp đồng có tồn tại không  if @maHD not in (select hd.MA\_HOP\_DONG  from HOPDONG hd  where hd.MA\_HOP\_DONG=@maHD)  begin  print N'Không tồn tại hợp đồng này'  rollback tran  end | **S(HOPDONG)**  //Xin khóa đọc bảng **HOPDONG**  (Xin thành công vì 2 khóa S tương thích nhau nên sẽ được phát) |
|  |  | B2.1: tạo biến @ngay\_hien\_tai và set bằng ngày cuối hợp đồng  declare @ngay\_hien\_tai date  set @ngay\_hien\_tai=( select top 1 hd.TG\_KET\_THUC  from HOPDONG hd  where hd.MA\_HOP\_DONG=@maHD) | **S(HOPDONG)**  //Xin khóa đọc bảng **HOPDONG**  (Xin thành công) |
|  |  | if @ngay\_hien\_tai is not null  begin  -- bắt đầu chờ  WAITFOR DELAY '00:00:15' |  |
| -- gia hạn ngày hợp đồng  set @ngay\_hien\_tai = (select DATEADD(DAY, @so\_ngay\_them, @ngay\_hien\_tai))  -- cập nhật vào hợp đồng  update HOPDONG  SET TG\_KET\_THUC = @ngay\_hien\_tai  WHERE MA\_HOP\_DONG = @maHD;  end | **X(HOPDONG)**  //Xin khóa ghi ở bảng **HOPDONG** để update gia hạn hợp đồng  (Không thành công chờ T2 giải phóng khóa S thì T2 mới có thể xin khóa X để ghi) |  |  |
| end try  begin catch  print N'Gia hạn hợp đồng thất bại'  rollback tran  end catch |  |  |  |
| COMMIT TRAN |  |  |  |
|  |  | -- gia hạn ngày hợp đồng  set @ngay\_hien\_tai = (select DATEADD(DAY, @so\_ngay\_them, @ngay\_hien\_tai))  -- cập nhật vào hợp đồng  update HOPDONG  SET TG\_KET\_THUC = @ngay\_hien\_tai  WHERE MA\_HOP\_DONG = @maHD;  end | **X(HOPDONG)**  //Xin khóa ghi ở bảng **HOPDONG** để update gia hạn hợp đồng  (Không thành công  Chờ T1 giải phóng khóa S thì T2 mới xin được khóa X để ghi) |
|  |  | end try  begin catch  print N'Gia hạn hợp đồng thất bại'  rollback tran  end catch  COMMIT TRAN |  |

### **Sinh viên thực hiện : Trần Thái San (20120177)**

#### Tình huống 1: DIRTY READ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ERR03: *DIRTY READ***  T1 (User = cửa hàng (đối tác)): thực cập nhật đơn giá món ăn  T2 (User = khách hàng): khách hàng thực hiện xem thông tin món ăn | | | | |
| **SP\_CAPNHAPGIAMONAN** |  | **SP\_XEMMONAN** |  |
| ***Input:*** @MAMONAN = ‘MA001’  @MA\_CUA\_HANG = ‘CH001’  @GIAMOI = 100000  ***Output***: 1: cập nhật thành công  0: cập nhật thất bại | ***Input***: *@MA\_MON\_AN = ‘MA001’*  ***Output***: : |
| SET TRANSACTION ISOLATION  LEVEL …. | SET TRANSACTION ISOLATION  LEVEL READ UNCOMMITTED |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra món ăn có tồn tại hay không  IF NOT EXISTS(SELECT \* FROM MON\_AN  WHERE @MAMONAN=MA\_MON\_AN  AND@MACH=MA\_CUA\_HANG)  BEGIN  PRINT N'KHÔNG TỒN TẠI MÓN ĂN'  ROLLBACK TRAN  RETURN 1  END | **S(MON\_AN)**  //Xin khoá đọc trên bảng MON\_AN với điều kiện  MaCH = ‘CH001’ và MaMonAn = ‘MA001’  (xin thành công) |  |  |
| B2: Thực hiện cập nhập giá mới  UPDATE MON\_AN  SET DON\_GIA=@GIAMOI  WHERE @MAMONAN=MA\_MON\_AN AND @MACH=MA\_CUA\_HANG | **U,X(MON\_AN)**  //Xin khoá U,X trên bảng MON\_AN với điều kiện @MAMONAN=’MA001’ @MACH = ‘CH001’ @GIAMOI =0  (xin thành công) |  |  |
| WAITFOR DELAY '00:00:10' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | Thực hiện xem món ăn của đối tác với MAMONAN = ‘MA001’  SELECT MA.MA\_MON\_AN, MA.TEN\_MON, MA.DON\_GIA, CH.TEN\_CUA\_HANG, MA.TINH\_TRANG, MA.TRANG\_THAI  FROM MON\_AN MA, CUAHANG CH  WHERE MA.MA\_MON\_AN=@MAMONAN  AND MA.MA\_CUA\_HANG=CH.MA\_CUA\_HANG | **S(MON\_AN)**  // Xin khoá đọc trên bảng MON\_AN với điều kiện  MA\_MON\_AN  = ‘MA001’  (xin thành công) |
|  |  | COMMIT TRAN |  |
| B3: Kiểm tra giá mới có bằng 0 hay không, nếu bằng thì rollbacktran  IF @GIAMOI=0  BEGIN  ROLLBACK TRAN  RETURN 1  END |  |  |  |
| COMMIT TRAN |  |  |  |

#### 

#### Tình huống 2: PHANTOM

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ERR03: *PHANTOM***  T1 (User = cửa hàng (đối tác)): thực hiện thêm một chi nhánh mới  T2 (User = khách hàng): thực hiện xem danh sách đối tác | | | | |
| **sp\_DT\_ThemChiNhanh** |  | **sp\_KH\_XemDSDoiTac** |  |
| ***Input:*** @machinhanh  @diachi  @sdt  @macuahang  @mahopdong  @khuvuc  ***Output***: 1: thêm thành công  0: thêm thất bại | ***Input***:  ***Output***: : 1: cập nhật thành công  0: cập nhật thất bại |
| SET TRANSACTION ISOLATION  LEVEL …. | SET TRANSACTION ISOLATION  LEVEL …… |
|  |  | BEGIN TRAN |  |
|  |  | B1: Xem danh sách đối tác(cửa hàng)  SELECT MA\_CUA\_HANG, TEN\_CUA\_HANG, EMAIL, THANH\_PHO, QUAN, SDT, SO\_CHI\_NHANH  FROM CUAHANG | **S(CUAHANG)**  //Xin khóa đọc trên bảng CUAHANG  (xin thành công) |
|  |  | WAITFOR DELAY '00:00:10' |  |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra trùng lắp của @machinhanh và @diachi  IF(EXISTS(SELECT \* FROM CHINHANH WHERE MA\_CUA\_HANG = @macuahang AND DIA\_CHI = @diachi))  begin  rollback tran  RETURN -1  end    IF(EXISTS(SELECT \* FROM CHINHANH WHERE MA\_CHI\_NHANH = @machinhanh))  begin  rollback tran  RETURN -1  end | **S(CHI\_NHANH)**  //Xin khoá đọc trên bảng CHI\_NHANH với điều kiện  @machinhanh = ‘CH001’  (xin thành công)  **S(CHI\_NHANH)**  //Xin khoá đọc trên bảng CHI\_NHANH với điều kiện  @diachi = ‘Dĩ An, Bình Dương’  (xin thành công) |  |  |
| B2: Thêm chi nhánh  INSERT INTO CHINHANH(MA\_CHI\_NHANH, SL\_DON\_MOINGAY, DIA\_CHI, SDT, TINH\_TRANG, MA\_CUA\_HANG, MA\_HOP\_DONG, KHUVUC)  VALUES(@machinhanh,NULL,@diachi, @sdt, null, @macuahang, @mahopdong, @khuvuc) | **X(CHI\_NHANH)**  //Xin khóa ghi trên bảng CHINHANH  (xin thành công) |  |  |
| B3: Cập nhập số lượng chi nhánh  UPDATE CUAHANG  SET SO\_CHI\_NHANH = SO\_CHI\_NHANH + 1  WHERE MA\_CUA\_HANG = @macuahang | **UX(CUAHANG)**  //Xin khóa ghi trên bảng CUAHANG  (xin thành công) |  |  |
| COMMIT TRAN |  |  |  |
|  |  | B1: Xem danh sách đối tác(cửa hàng)  SELECT MA\_CUA\_HANG, TEN\_CUA\_HANG, EMAIL, THANH\_PHO, QUAN, SDT, SO\_CHI\_NHANH  FROM CUAHANG | **S(CUAHANG)**  //Xin khóa đọc trên bảng CUAHANG  (xin thành công) |
|  |  | COMMIT TRAN |  |

### **Sinh viên thực hiện: Nguyễn Nhật Trường (20120229)**

#### Tình huống 1: UNREPEATABLE READ

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: UNREPEATABLE READ**  T1 (User = khách hàng): xem tình trạng đơn hàng  T2 (User = tài xế): tài xế thực hiện cập nhật tình trạng đơn hàng | | | |
| **USP\_KIEMTRA\_TTGH** | **Khóa** | **USP\_TX\_CAPNHAT\_TTGH** | **Khóa** |
| ***Input:*** @MA\_KHACH\_HANG CHAR(5), @MA\_DON CHAR(5) ***Output***: ……. | ***Input***: @MA\_TAI\_XE CHAR(5), @MA\_DON CHAR(5), @TTGH NVARCHAR(255)  ***Output***: |
| BEGIN TRAN |  |  |  |
| B1: kiểm tra @MA\_DON\_HANG tồn tại    IF NOT EXISTS(SELECT DDH.\* FROM DONDATHANG DDH WHERE DDH.MA\_KHACH\_HANG=@MA\_KHACH\_HANG AND DDH.MA\_DON=@MA\_DON)  BEGIN  PRINT N'ĐH không tồn tại'  RETURN  END | **S(DONDATHANG)** |  |  |
| DECLARE @TTGH NVARCHAR(255)  SET @TTGH = 0  SET @TTGH = (SELECT TINH\_TRANG FROM DONDATHANG WHERE MA\_DON = @MA\_DON) | **S(DONDATHANG)** |  |  |
| WAITFOR DELAY '00:00:015' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | UPDATE DONDATHANG  SET TINH\_TRANG = @TTGH  WHERE MA\_DON = @MA\_DON AND MA\_TAI\_XE = @MA\_TAI\_XE | **U.X(DONDATHANG)** |
|  |  | COMMIT TRAN |  |
| IF ((SELECT TINH\_TRANG FROM DONDATHANG WHERE MA\_DON = @MA\_DON) <> @TTGH)  BEGIN  RAISERROR(N'Dữ liệu hai lần đọc khác nhau',16,1)  END  SELECT \*  FROM DONDATHANG  WHERE MA\_DON = @MA\_DON | **S(DONDATHANG)** |  |  |
| COMMIT TRAN |  |  |  |

#### Tình huống 2: DIRTY READ

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: DIRTY READ**  T1 (User = Đối tác (cửa hàng)): Đối tác thực hiện cập nhật giá của món ăn B (@MA\_MON\_AN = ‘MA002’)  T2 (User = Khách hàng): Khách hàng xem giá món ăn B ((@MA\_MON\_AN = ‘MA002’) | | | |
| **SP\_DT\_CAPNHAT\_GIA\_MONAN** | **Khóa** | **USP\_KH\_XEMGIA\_MONAN** | **Khóa** |
| ***Input:*** @MA\_MON\_AN CHAR(5), @GIA\_UPDATE MONEY  ***Output***: ……. | ***Input***: @MA\_MON\_AN CHAR(5)  ***Output***: |
| BEGIN TRAN |  |  |  |
| B1: kiểm tra @MA\_MON\_AN tồn tại    IF NOT EXISTS (SELECT \* FROM MON\_AN WHERE MA\_MON\_AN=@MA\_MON\_AN)  BEGIN  PRINT N'MÓN ĂN KHÔNG TỒN TẠI'  RETURN  END | **S(MON\_AN)** |  |  |
| DECLARE @GIASP MONEY  SET @GIASP = (SELECT DON\_GIA FROM MON\_AN WHERE MA\_MON\_AN=@MA\_MON\_AN)  UPDATE MON\_AN  SET DON\_GIA = @GIA\_UPDATE  WHERE MA\_MON\_AN=@MA\_MON\_AN | **U,X(MON\_AN)** |  |  |
| WAITFOR DELAY '00:00:05' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: kiểm tra @MA\_MON\_AN tồn tại  IF NOT EXISTS (SELECT \* FROM MON\_AN WHERE MA\_MON\_AN=@MA\_MON\_AN)  BEGIN  PRINT N'MÓN ĂN KHÔNG TỒN TẠI'  RETURN  END | **S(MON\_AN)** |
|  |  | SELECT \*  FROM MON\_AN WITH(NOLOCK)  WHERE MA\_MON\_AN=@MA\_MON\_AN | **S(MON\_AN)** |
|  |  | COMMIT TRAN |  |
| IF (@GIA\_UPDATE = 0)  BEGIN  ROLLBACK TRAN  RETURN  END |  |  |  |
|  |  |  |  |
| COMMIT TRAN |  |  |  |

#### Tình huống 3: CIRCLE DEADLOCK

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: CIRCLE DEADLOCK**  T1: Người quản trị 1 cập nhật thông tin tài khoản của khách hàng (đổi mật khẩu) và cập nhật thông tin của tài xế (Email).  T2: Quản trị 2 cập nhật thông tin của tài xế(Email) và cập nhật thông tin tài khoản của khách hàng (đổi mật khẩu). | | | |
| **USP\_QT1\_CAPNHAT\_TT @TEN\_DANG\_NHAP** | **Khóa** | **USP\_QT2\_CAPNHAT\_TT** | **Khóa** |
| ***Input:*** @TEN\_DANG\_NHAP NVARCHAR(255), @MAT\_KHAU\_MOI CHAR(100), @MATX CHAR(5), @EMAIL\_MOI CHAR(100)  ***Output***: ……. | ***Input***: @TEN\_DANG\_NHAP NVARCHAR(255), @MAT\_KHAU\_MOI CHAR(100), @MATX CHAR(5), @EMAIL\_MOI CHAR(100)  ***Output***: |
| BEGIN TRAN |  |  |  |
| B1: kiểm tra @TEN\_DANG\_NHAP tồn tại    IF NOT EXISTS(SELECT TK.\* FROM TAIKHOAN TK WHERE TEN\_DANG\_NHAP=@TEN\_DANG\_NHAP)  BEGIN  PRINT N'Tài khoản không tồn tại'  RETURN  END | **S(TAIKHOAN)** |  |  |
| UPDATE TAIKHOAN SET MAT\_KHAU = @MAT\_KHAU\_MOI WHERE TEN\_DANG\_NHAP=@TEN\_DANG\_NHAP | **U,X(TAIKHOAN)** |  |  |
| WAITFOR DELAY '00:00:05' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1: kiểm tra @TEN\_DANG\_NHAP tồn tại    IF NOT EXISTS(SELECT TK.\* FROM TAIKHOAN TK WHERE TEN\_DANG\_NHAP=@TEN\_DANG\_NHAP)  BEGIN  PRINT N'Tài khoản không tồn tại'  RETURN  END | **S(TAIKHOAN)** |
|  |  | UPDATE TAIXE  SET EMAIL = @EMAIL\_MOI  WHERE MA\_TAI\_XE = @MATX | **U,X(TAIXE)** |
|  |  | WAITFOR DELAY '00:00:05' |  |
|  |  | UPDATE TAIKHOAN SET MAT\_KHAU = @MAT\_KHAU\_MOI WHERE TEN\_DANG\_NHAP=@TEN\_DANG\_NHAP | **S(TAIKHOAN)** |
| UPDATE TAIXE  SET EMAIL = @EMAIL\_MOI  WHERE MA\_TAI\_XE = @MATX | **U,X(TAIXE)** |  |  |
| COMMIT TRAN |  |  |  |
|  |  | COMMIT TRAN |  |

# XỬ LÍ TÌNH HUỐNG TRANH CHẤP

1. **Sinh viên thực hiện : Trịnh Hữu Hiệp (20120079)**

### Tình huống 1: LOST UPDATE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ERR01: *lost update***  T1 (User = cửa hàng (đối tác)): đối tác thực hiện nhập xuất số lượng món ăn, cập nhật số lượng món ăn tại một chi nhánh  T2 (User = cửa hàng (đối tác)): đối tác thực hiện thêm mới đơn hàng (khi khách hàng mua tại chi nhánh) | | | | |
| **SP\_CH\_CAP\_NHAT\_SL\_MON\_AN** |  | **SP\_KHACH\_HANG\_DAT\_HANG\_MON\_AN** |  |
| ***Input:*** @MA\_CHI\_NHANH = ‘CN001’  @MA\_MON\_AN = ‘MA001’  @SO\_LUONG\_THEM = 100  ***Output***: 1: cập nhật thành công  0: cập nhật thất bại | ***Input***: @MA\_DON = ‘DH001’  *@MA\_MON\_AN = ‘MA001’*  *@SO\_LUONG = 10*  *@DIA\_CHI = ‘QUANG NAM’*  *@PHI\_VAN\_CHUYEN = 10000*  *@MA\_KHACH\_HANG = ‘KH001’*  *@MA\_TAI\_XE = ‘TX001’*  *@MA\_CHI\_NHANH = ‘CN001’*  ***Output***: 1: cập nhật thành công  0: cập nhật thất bại |
| SET TRANSACTION ISOLATION  LEVEL *REPEATABLE READ* |  |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra thông tin tồn tại của @MA\_CHI\_NHANH và @MA\_MON\_AN  IF NOT EXISTS  (SELECT \* FROM MON\_AN\_CHI\_NHANH WHERE MA\_CHI\_NHANH = @MA\_CHI\_NHANH AND MA\_MON\_AN = @MA\_MON\_AN) --(1)  BEGIN  PRINT @MA\_CHI\_NHANH +', '+  @MA\_MON\_AN + N' KHÔNG HỢP LỆ !'  ROLLBACK TRAN  RETURN 0  END | **S(MON\_AN\_CHI\_**  **NHANH)**  //Xin khoá đọc trên bảng MON\_AN\_  CHI\_NHAnH  // (xin thành công)  Giữ các khóa đọc đến cuối giao tác do mức cô lập RR |  |  |
| B2: Kiểm tra @SO\_LUONG\_THEM  IF @SO\_LUONG\_THEM <1  BEGIN  PRINT N'SỐ LƯỢNG THÊM KHÔNG HỢP LỆ !'  ROLLBACK TRAN  RETURN 0  END | //kiểm tra @SO\_LUONG\_THEM hợp lệ |  |  |
| DECLARE @SO\_LUONG\_HIEN\_CO INT  SET @SO\_LUONG\_HIEN\_CO =  (SELECT SELECT SO\_LUONG FROM MON\_AN\_CHI\_NHANH WHERE MA\_CHI\_NHANH = @MA\_CHI\_NHANH AND MA\_MON\_AN = @MA\_MON\_AN) BEGIN  PRINT ' SỐ LƯỢNG HIỆN CÓ KHÔNG HỢP LỆ !'  ROLLBACK TRAN  RETURN 0  END | **S(MON\_AN\_CHI**  **\_NHANH)**  //không cần xin khoá đọc trên bảng MON\_AN\_  CHI\_NHANH  Do trước đó đã xin rồi |  |  | |
| WAITFOR DELAY '00:00:10' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1.1: Kiểm tra thông tin tồn tại của @MA\_KHACH\_HANG ,@MA\_TAI\_XE, @MA\_CHI\_NHANH  IF( @MA\_KHACH\_HANG NOT IN( SELECT MA\_KHACH\_HANG FROM KHACHHANG) OR  @MA\_TAI\_XE NOT IN( SELECT MA\_TAI\_XE FROM TAIXE) OR @MA\_CHI\_NHANH NOT IN  ( SELECT MA\_CHI\_NHANH FROM CHINHANH))  BEGIN  PRINT N'KHÔNG TỒN TẠI ' + @MA\_KHACH\_HANG +N' HOẶC '+ @MA\_TAI\_XE +N' HOẶC '+ @MA\_CHI\_NHANH  ROLLBACK TRAN  RETURN 0  END | **S(KHACHHANG)**  **S(TAIXE)**  **S(CHINHANH)**  //Xin khoá đọc trên bảng KHACHHANG,TAIXE,  CHINHANH  (xin thành công)  Giữ các khóa đọc đến cuối giao tác do mức cô lập RR |
|  |  | B1.2: Kiểm tra thông tin tồn tại của @MA\_DON và @MA\_MON\_AN  IF(EXISTS( SELECT \* FROM DONDATHANG DH WHERE DH.MA\_DON = @MA\_DON))  BEGIN  PRINT @MA\_DON + N' ĐÃ TỒN TẠI'  ROLLBACK TRAN  RETURN 0  END  IF(@MA\_MON\_AN NOT IN  (SELECT MA\_MON\_AN FROM MON\_AN WHERE TINH\_TRANG = N'có bán'))  BEGIN  PRINT @MA\_MON\_AN + N' KHÔNG TỒN TẠI'  ROLLBACK TRAN  RETURN 0  END | **S(DON\_DAT\_HANG)**  //Xin khoá đọc trên bảng (xin thành công) DON\_DAT\_HANG  Giữ các khóa đọc đến cuối giao tác do mức cô lập RR  **S(MON\_AN)**  //Xin khoá đọc trên bảng MON\_AN  (xin thành công)  Giữ các khóa đọc đến cuối giao tác do mức cô lập RR |
|  |  | B2: Kiểm tra thông tin tồn tại @SO\_lUONG  IF (@SO\_LUONG > ( SELECT A.SO\_LUONG FROM MON\_AN\_CHI\_NHANH A WHERE A.MA\_CHI\_NHANH = @MA\_CHI\_NHANH AND A.MA\_MON\_AN =@MA\_MON\_AN)) --(2)  BEGIN  PRINT N'SỐ LƯỢNG HIỆN TẠI CỦA CHI NHÁNH '+ @MA\_CHI\_NHANH+' KHÔNG ĐỦ ĐÁP ỨNG'  ROLLBACK TRAN  RETURN 0  END | **S(MON\_AN\_CHI\_NHANH)**  //Xin khoá đọc trên bảng MON\_AN\_CHI  \_NHANH  *(xin không thành công)*  Do T1 đã giữ khóa đọc trên MON\_AN\_CHI  \_NHANH và chưa commit  Chờ **T1** Nhả khóa  (sau Khi **COMMIT**) |
|  |  | B3: lấy đơn giá món ăn , tạo biến tổng tiền trước khi insert  DECLARE @DON\_GIA\_MON FLOAT  SET @DON\_GIA\_MON = (SELECT DON\_GIA FROM MON\_AN WHERE MA\_MON\_AN = @MA\_MON\_AN)  DECLARE @TONG\_TIEN FLOAT  SET @TONG\_TIEN = @DON\_GIA\_MON \* @SO\_LUONG | **S(MON\_AN)**  //Xin khoá đọc trên bảng MON\_AN  (xin thành công) |
|  |  | B4: insert dữ liệu vào bảng DONDATHANG VÀ CHITIET\_DONHANH  -- INSERT DON DAT HÀNG  INSERT INTO DONDATHANG VALUES(@MA\_DON,@TONG\_TIEN,@DIA\_CHI,N'chờ xác nhận', @PHI\_VAN\_CHUYEN, @MA\_KHACH\_HANG,@MA\_TAI\_XE,@MA\_CHI\_NHANH,1)  -- INSERT CHI TIẾT ĐƠN HÀNG  INSERT INTO CHITIET\_DONHANG VALUES(@MA\_DON , @MA\_MON\_AN, @SO\_LUONG, @TONG\_TIEN) | **X(DONDATHANG)**  //Xin khoá ghi trên bảng DONDATHANG  (xin thành công)  **X(CHITIET\_**  **DON\_HANG)**  //Xin khoá ghi trên bảng CHITIET\_DONHANG  (xin thành công) |
|  |  | B5: cập nhật số lượng món ăn tại chi nhánh khách hàng mua hàng (‘CN001)  UPDATE MON\_AN\_CHI\_NHANH  SET SO\_LUONG = SO\_LUONG - @SO\_LUONG --(3)  WHERE @MA\_CHI\_NHANH = MA\_CHI\_NHANH AND @MA\_MON\_AN = MA\_MON\_AN | **X(MON\_AN**  **\_CHI\_NHANH)**  //Xin khoá GHI trên bảng MON\_AN\_  CHI\_NHANH  (xin thành công) |
|  |  | COMMIT TRAN  PRINT N'CẬP NHẬT THÀNH CÔNG'  RETURN 1 |  |
| B3:cập nhật số lượng món ăn tại chi nhánh  UPDATE MON\_AN\_CHI\_NHANH  SET SO\_LUONG = @SO\_LUONG\_HIEN\_CO + @SO\_LUONG\_THEM  WHERE MA\_CHI\_NHANH = @MA\_CHI\_NHANH AND MA\_MON\_AN = @MA\_MON\_AN | **X(MON\_AN**  **\_CHI\_NHANH)**  //Xin khoá ghi trên bản MON\_AN\_CHI  \_NHANH  (xin thành công)  Giữ khóa đến hết giao tác |  |  |
| COMMIT TRAN  PRINT N'CẬP NHẬT THÀNH CÔNG'  RETURN 1 | **Nhả khóa trên các đơn vị dữ liệu** |  |  |

### Tình huống 2: UNREPEATABLE READ

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: UNREPEATABLE READ**  T1 (User = cửa hàng(đối tác)): đối tác thực hiện cập nhật đơn giá của món ăn (@MA\_MON\_AN = ‘MA001’)  T2 (User = khách hàng): xem tổng tiền của một món ăn (@MA\_MON\_AN = ‘MA001’) với số lượng nhất định | | | |
| **SP\_KH\_XEM\_TONG\_TIEN\_MOT\_MON\_AN** | **Khóa** | **SP\_CH\_CAP\_NHAT\_DON\_GIA\_MOT\_MON\_AN** | **Khóa** |
| ***Input:*** @MA\_CUA\_HANG = ‘CH001’  @MA\_MON\_AN = ‘MA001’  @SOUONG = 10  ***Output***: 1: hiển thị thành công  0: thất bại | ***Input***: @MA\_CUA\_HANG = ‘CH001’  @MA\_MON\_AN = ‘MA001’  *@NEW\_DONGIA = 100000*  ***Output***: 1: cập nhật thành công  0: cập nhật thất bại |
| SET TRANSACTION ISOLATION  LEVEL **REPEATABLE READ** |  |
| BEGIN TRAN |  |  |  |
| B1: kiểm tra @MA\_CUA\_HANG tồn tại  IF(@MA\_CUA\_HANG NOT IN ( SELECT MA\_CUA\_HANG FROM CUAHANG ))  BEGIN  PRINT @MA\_CUA\_HANG + N' KHÔNG HỢP LỆ !'  ROLLBACK TRAN  RETURN 0  END | **S(CUAHANG)**  //Xin khoá đọc trên bảng  (xin thành công) giữ khóa đến hết giao tác do mức cô lập RR |  |  |
| B1: kiểm tra @MA\_MON\_AN tồn tại và thuộc về của hàng ‘CH001’  IF(NOT EXISTS ( SELECT \* FROM MON\_AN MA WHERE MA.MA\_MON\_AN = @MA\_MON\_AN AND MA.TINH\_TRANG = N'có bán' AND MA.TRANG\_THAI = 1 AND MA.MA\_CUA\_HANG = @MA\_CUA\_HANG))  BEGIN  PRINT @MA\_MON\_AN + N' KHÔNG HỢP LỆ !'  ROLLBACK TRAN  RETURN 0  END | **S(MON\_AN)**  //Xin khoá đọc trên bảng MON\_AN  (xin thành công)  giữ khóa đến hết giao tác do mức cô lập  RR |  |  |
| DECLARE @DONGIA FLOAT  SET @DONGIA = (SELECT DON\_GIA FROM MON\_AN WHERE MA\_MON\_AN = @MA\_MON\_AN)  PRINT N'Giá Món ăn là : '+ CAST(@DONGIA AS  NVARCHAR(30)) | **S(MON\_AN)**  Không cần xin khóa do trước đó đã xin rồi |  |  |
| WAITFOR DELAY '00:00:15' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | IF(@MA\_CUA\_HANG NOT IN ( SELECT MA\_CUA\_HANG FROM CUAHANG))  BEGIN  PRINT @MA\_CUA\_HANG + N' KHÔNG HỢP LỆ !'  ROLLBACK TRAN  RETURN 0  END | **S(CUAHANG)**  // Xin khoá đọc trên bảng CUAHANG  (xin không thành công)  *Chờ* ***T1*** *nhả khóa ( Sau khi COMMIT ) do khóa được giữ đến hết giao tác )* |
|  |  | IF(NOT EXISTS ( SELECT \* FROM MON\_AN MA WHERE MA.MA\_MON\_AN = @MA\_MON\_AN AND TRANG\_THAI !=0 AND MA.MA\_CUA\_HANG = @MA\_CUA\_HANG))  BEGIN  PRINT @MA\_MON\_AN + N' KHÔNG HỢP LỆ !'  ROLLBACK TRAN  RETURN 0  END --(1) | **S(MON\_AN)**  // Xin khoá đọc trên bảng MON\_AN  (xin thành công) |
|  |  | UPDATE MON\_AN  SET DON\_GIA = @NEW\_DONGIA  WHERE MA\_MON\_AN = @MA\_MON\_AN1 | **X(MON\_AN)**  //Xin khoá ghi trên bảng MON\_AN  (xin thành công) |
|  |  | COMMIT TRAN  PRINT N'CẬP NHẬT THÀNH CÔNG'  RETURN 1 |  |
| DECLARE @TONGTIEN FLOAT  SET @TONGTIEN = (SELECT @SO\_LUONG\* DON\_GIA FROM MON\_AN WHERE @MA\_MON\_AN = MA\_MON\_AN)  PRINT N'Tổng tiền món ăn với số lượng : ' +  CAST(@SO\_LUONG AS NVARCHAR(30)) + N' là : '+  CAST(@TONGTIEN AS NVARCHAR(30)) | **X(MON\_AN)**  //Xin khoá ghi trên bảng MON\_AN (xin thành công do T2 đã commit và nhả khóa ghi trên bảng MON\_AN)  Giữ khóa đến hết giao tác |  |  |
| COMMIT TRAN  PRINT N'HIỂN THỊ THÀNH CÔNG'  RETURN 1 | T1 Nhả các khóa và T2 thực hiện các bước sau đó |  |  |

## **Sinh viên thực hiện: Nguyễn Thị Hồng Nhung (20120093)**

### Tình huống 1: Lỗi Unrepeatable Read

|  |
| --- |
| Mô tả cách sửa lỗi: Đặt mức cô lập **REPEATABLE READ** cho T1, Tạo Shared Lock trên đơn vị dữ liệu được đọc và T1 giữ shared lock này đến hết giao tác, giao tác T2 phải chờ đến khi giao tác T1kết thúc nếu muốn cập nhật, thay đổi giá trị trên đơn vị dữ liệu này. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ERR01: UNREPEATABLE READ**  T1 (User = Khách hàng): khách hàng A thực hiện đăng nhập  T2 (User = Khách hàng): ở thiết bị khác của khách hàng A lại cập nhật mật khẩu | | | | |
| **dang\_nhap** | **Khóa** | **cap\_nhat\_mat\_khau** | **Khóa** | |
| ***Input: tên đăng nhập (@ten\_tk)***  ***Mật khẩu (@mat\_khau)***  ***Output***: nếu thành công: output ra table gồm:  loại tài khoản, mã khách hàng | ***Input***: ***tên đăng nhập (@ten\_tk)***  ***Mật khẩu (@mat\_khau)***  ***Mật khẩu mới (@mat\_khau\_moi)***  ***Output***: nếu thành công mật khẩu mới được cập nhật, nếu không thành công rollback |
| SET TRANSACTION ISOLATION  LEVEL **REPEATABLE READ** | SET TRANSACTION ISOLATION  LEVEL **READ COMMITTED** |
| BEGIN TRAN |  |  |  | |
| B1: Kiểm tra tên đăng nhập có tồn tại không  if not exists (select tk.TEN\_DANG\_NHAP from TAIKHOAN tk where tk.TEN\_DANG\_NHAP=@ten\_tk)  begin  print N'Không tồn tại tên đăng nhập'  rollback tran  return -1  end | **S(TAIKHOAN)**  //Xin khóa đọc trên bảng **TAIKHOAN**  sau khi đọc xong thì giữ khóa đến khi kết thúc giao tác  (Xin thành công) |  |  | |
| B2: Kiểm tra mật khẩu có đúng không  if not exists (select tk.TEN\_DANG\_NHAP from TAIKHOAN tk where tk.TEN\_DANG\_NHAP=@ten\_tk and tk.MAT\_KHAU= @mat\_khau)  begin  print N'Sai mật khẩu'  rollback tran  return -1  end |  |  | |
| declare @loai\_tk int  set @loai\_tk = NULL  SET @loai\_tk = (SELECT TOP 1 tk.LOAI\_TK  FROM TAIKHOAN tk  WHERE tk.TEN\_DANG\_NHAP = @ten\_tk AND tk.MAT\_KHAU = @mat\_khau AND tk.TRANG\_THAI = 1) |  |  |
| WAITFOR DELAY '00:00:10' |  |  |  | |
|  |  | BEGIN TRAN |  | |
|  |  | B1: Kiểm tra thông tin  if not exists (select tk.TEN\_DANG\_NHAP from TAIKHOAN tk where tk.TEN\_DANG\_NHAP=@ten\_tk)  begin  print N'Sai tên đăng nhập'  rollback tran  return -1  end | **S(TAIKHOAN)**  **/**/Xin khóa đọc trên bảng **TAIKHOAN**  Vì khóa S tương thích nhau nên vẫn phát khóa S, sau khi đọc xong sẽ giải phóng khóa(Xin thành công) | |
|  |  | B2: Kiểm tra mật khẩu  if not exists ( select tk.TEN\_DANG\_NHAP from TAIKHOAN tk where tk.TEN\_DANG\_NHAP=@ten\_tk and tk.MAT\_KHAU= @mat\_khau)  begin  print N'Sai mật khẩu. Vui lòng nhập đúng mật khẩu để sửa đổi thông tin'  rollback tran  return -1  end | **S(TAIKHOAN)**  **/**/Xin khóa đọc trên bảng **TAIKHOAN**  Vì khóa S tương thích nhau nên vẫn phát khóa S, sau khi đọc xong sẽ giải phóng khóa (Xin thành công) | |
|  |  |  | **X(TAIKHOAN)**  //Xin khóa ghi trên bảng **TAIKHOAN**  (Chờ T1 commit thì mới có thể xin khóa X để ghi vì T1 đang giữ khóa S đến hết giao tác)  (Xin chưa thành công) Đang chờ | |
| IF @loai\_tk IS NOT NULL  begin  if @loai\_tk=0  begin  SELECT distinct @loai\_tk AS 'loai\_tk', CH.MA\_CUA\_HANG AS 'ma'  FROM CUAHANG CH  WHERE CH.MA\_CUA\_HANG = @ten\_tk  COMMIT TRAN  RETURN  end  else if @loai\_tk=1  BEGIN  SELECT distinct @loai\_tk AS 'loai\_tk', KH.MA\_KHACH\_HANG AS 'ma'  FROM KHACHHANG KH, TAIKHOAN tk  WHERE KH.TEN\_DANG\_NHAP = @ten\_tk and tk.MAT\_KHAU=@mat\_khau  COMMIT TRAN  RETURN  END  else if @loai\_tk=2  BEGIN  SELECT distinct @loai\_tk AS 'loai\_tk', TX.MA\_TAI\_XE AS 'ma'  FROM TAIXE TX, TAIKHOAN tk  WHERE TX.TEN\_DANG\_NHAP = @ten\_tk and tk.MAT\_KHAU=@mat\_khau  COMMIT TRAN  RETURN  END  else if @loai\_tk=3  begin  select distinct @loai\_tk as 'loai\_tk', NV.MA\_NHAN\_VIEN from NHANVIEN NV, TAIKHOAN tk  where NV.TEN\_DANG\_NHAP=@ten\_tk and tk.MAT\_KHAU=@mat\_khau  end  else if @loai\_tk=3  begin  declare @ad char(10)  set @ad= 'admin'  select distinct @loai\_tk as 'loai\_tk', tk.TEN\_DANG\_NHAP from TAIKHOAN tk  where tk.TEN\_DANG\_NHAP=@ten\_tk and tk.MAT\_KHAU=@mat\_khau  end  end | **S(TAIKHOAN)**  //Xin khóa đọc trên bảng **TAIKHOAN**  Giữ khóa **S** đến hết giao tác  (Xin thành công) |  |  |
| end try  begin catch  print N'Đăng nhập thất bại'  rollback tran  end catch  commit tran  go |  |  |  |
| COMMIT TRAN | Giải phóng khóa **S** |  |  |
|  |  | update TAIKHOAN  set MAT\_KHAU=@mat\_khau\_moi  where TEN\_DANG\_NHAP= @ten\_tk  print N'Đổi mật khẩu thành công'  end try  begin catch  print N'Thay đổi mật khẩu thất bại'  rollback tran  end catch | **X(TAIKHOAN)**  //Xin khóa ghi trên bảng TAIKHOAN(xin thành công do T1 đã commit và giải phóng khóa S) | |
|  |  | COMMIT TRAN | Giải phóng khóa **X** | |

### Tình huống 2: Lỗi Conversion Deadlock

|  |
| --- |
| Mô tả sửa lỗi: Đối với trường hợp **CONVERSION DEADLOCK**, thực hiện đề xuất sử dụng khóa tăng cấp (Update lock) để giải quyết tình trạng lỗi này. Khi thực hiện, T1 sẽ được cấp khóa tăng cấp, khóa này chỉ cho phép đọc dữ liệu trên bảng HOPDONG, đồng thời T2 cũng thực hiện xin khóa tăng cấp. Tuy nhiên, T2 phải chờ T1 nhả khóa thì mới có thể thực hiện các thao tác của mình, vì do T2 cũng xin khóa tăng cấp cùng đơn vị dữ liệu với T1. Sau đó, T1 sẽ update lên khóa ghi để thực hiện cập nhật dữ liệu và commit. Lúc này, T2 bắt đầu thực hiện tương tự như T1. |

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR02: CONVERSION DEADLOCK**  T1 (User = Nhân viên): nhân viên A thực hiện gia hạn hợp đồng của cửa hàng C  T2 (User = Nhân viên): nhân viên B cũng thực hiện gia hạn hợp đồng của cửa hàng C | | | |
| **Gia\_han\_hop\_dong** | **Khóa** | **Gia\_han\_hop\_dong** | **Khóa** |
| ***Input:*** Mã hợp đồng A (*@maHD*), số lượng ngày thêm gia hạn hợp đồng (*@so\_ngay\_them)*  ***Output***: thời gian kết thúc được gia hạn thêm với số ngày gia hạn hợp đồng thêm | ***Input:*** Mã hợp đồng A(*@maHD*), số lượng ngày thêm gia hạn hợp đồng (*@so\_ngay\_them)*  ***Output***: thời gian kết thúc được gia hạn thêm với số ngày gia hạn hợp đồng thêm |
| SET TRANSACTION ISOLATION LEVEL **SERIALIZABLE** | SET TRANSACTION ISOLATION LEVEL **SERIALIZABLE** |  |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra mã Hợp đồng có tồn tại không  if @maHD not in (select hd.MA\_HOP\_DONG  from HOPDONG hd with (UPDLOCK)  where hd.MA\_HOP\_DONG=@maHD)  begin  print N'Không tồn tại hợp đồng này'  rollback tran  end | **U(HOPDONG)**  //Xin khóa update lock trên bảng  **HOPDONG**  (Xin thành công)  Giữ đến cuối giao tác |  |  |
| B2: tạo biến @ngay\_hien\_tai và set bằng ngày cuối hợp đồng  declare @ngay\_hien\_tai date  set @ngay\_hien\_tai=( select top 1 hd.TG\_KET\_THUC  from HOPDONG hd  where hd.MA\_HOP\_DONG=@maHD) |  |  |
| if @ngay\_hien\_tai is not null  begin  -- bắt đầu chờ  WAITFOR DELAY '00:00:15' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | B1.1: Kiểm tra mã Hợp đồng có tồn tại không  if @maHD not in (select hd.MA\_HOP\_DONG  from HOPDONG hd  where hd.MA\_HOP\_DONG=@maHD)  begin  print N'Không tồn tại hợp đồng này'  rollback tran  end | **U(HOPDONG)**  //Xin khóa tăng cấp trên bảng **HOPDONG**  Tuy nhiên vì T1 đang  giữ khóa tăng cấp trên bảng **HOPDONG** nên  T2 phải chờ T1 giải phóng khóa. Sau khi T1 commit thì T2 sẽ xin thành công khóa **U** |
|  |  | B2.1: tạo biến @ngay\_hien\_tai và set bằng ngày cuối hợp đồng  declare @ngay\_hien\_tai date  set @ngay\_hien\_tai=( select top 1 hd.TG\_KET\_THUC  from HOPDONG hd  where hd.MA\_HOP\_DONG=@maHD) |
|  |  | if @ngay\_hien\_tai is not null  begin  -- bắt đầu chờ  WAITFOR DELAY '00:00:15' |
| -- gia hạn ngày hợp đồng  set @ngay\_hien\_tai = (select DATEADD(DAY, @so\_ngay\_them, @ngay\_hien\_tai))  -- cập nhật vào hợp đồng  update HOPDONG  SET TG\_KET\_THUC = @ngay\_hien\_tai  WHERE MA\_HOP\_DONG = @maHD;  end | **X(HOPDONG)**  //T1 update khóa  **U** thành khóa **X** để  có thể ghi trên  bảng **HOPDONG** và giữ khóa **X** đến cuối giao tác(Thành công) |  |  |
| end try  begin catch  print N'Gia hạn hợp đồng thất bại'  rollback tran  end catch |  |  |  |
| COMMIT TRAN | Giải phóng khóa **X** |  |  |
|  |  | -- gia hạn ngày hợp đồng  set @ngay\_hien\_tai = (select DATEADD(DAY, @so\_ngay\_them, @ngay\_hien\_tai))  -- cập nhật vào hợp đồng  update HOPDONG  SET TG\_KET\_THUC = @ngay\_hien\_tai  WHERE MA\_HOP\_DONG = @maHD;  end | **X(HOPDONG)**  //T1 update khóa  **U** thành khóa **X** để  có thể ghi trên  bảng **HOPDONG** và giữ khóa **X** đến cuối giao tác (Thành công) |
|  |  | end try  begin catch  print N'Gia hạn hợp đồng thất bại'  rollback tran  end catch |  |
|  |  | COMMIT TRAN | Giải phóng khóa **X** |

## **3. Sinh viên thực hiện: Trần Thái San**

### Tình huống 1: DIRTY READ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ERR03: *Dirty Read***  T1 (User = cửa hàng (đối tác)): thực cập nhật đơn giá món ăn  T2 (User = khách hàng): khách hàng thực hiện xem thông tin món ăn | | | | |
| **SP\_CAPNHAPGIAMONAN** |  | **SP\_XEMMONAN** |  |
| ***Input:*** @MAMONAN = ‘MA001’  @MA\_CUA\_HANG = ‘CH001’  @GIAMOI = 100000  ***Output***: 1: cập nhật thành công  0: cập nhật thất bại | ***Input***: *@MA\_MON\_AN = ‘MA001’*  ***Output***: : |
|  |  |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra món ăn có tồn tại hay không  IF NOT EXISTS(SELECT \* FROM MON\_AN  WHERE @MAMONAN=MA\_MON\_AN  AND@MACH=MA\_CUA\_HANG)  BEGIN  PRINT N'KHÔNG TỒN TẠI MÓN ĂN'  ROLLBACK TRAN  RETURN 1  END | **S(MON\_AN)**  //Xin khoá đọc trên bảng MON\_AN với điều kiện  MaCH = ‘CH001’ và MaMonAn = ‘MA001’  (xin thành công) |  |  |
| B2: Thực hiện cập nhập giá mới  UPDATE MON\_AN  SET DON\_GIA=@GIAMOI  WHERE @MAMONAN=MA\_MON\_AN AND @MACH=MA\_CUA\_HANG | **U,X(MON\_AN)**  //Xin khoá U,X trên bảng MON\_AN với điều kiện @MAMONAN=’MA001’ @MACH = ‘CH001’ @GIAMOI =0  (xin thành công) |  |  |
| WAITFOR DELAY '00:00:10' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | Thực hiện xem món ăn của đối tác với MAMONAN = ‘MA001’  SELECT MA.MA\_MON\_AN, MA.TEN\_MON, MA.DON\_GIA, CH.TEN\_CUA\_HANG, MA.TINH\_TRANG, MA.TRANG\_THAI  FROM MON\_AN MA, CUAHANG CH  WHERE MA.MA\_MON\_AN=@MAMONAN  AND MA.MA\_CUA\_HANG=CH.MA\_CUA\_HANG | **S(MON\_AN)**  // Xin khoá đọc trên bảng MON\_AN với điều kiện  MA\_MON\_AN  = ‘MA001’  (xin thành công) |
|  |  | COMMIT TRAN |  |
| B3: Kiểm tra giá mới có bằng 0 hay không, nếu bằng thì rollbacktran  IF @GIAMOI=0  BEGIN  ROLLBACK TRAN  RETURN 1  END |  |  |  |
| COMMIT TRAN |  |  |  |

GIẢI THÍCH: Để xử lý Dirty Read, nhóm em đề xuất mức cô lập mặc định cho T2 ( Read Committed ) khi đó khi T1 thực hiện select và update giá trên bảng MON\_AN thì T2 cũng thực hiện đọc giá sản phẩm rồi tiếp tục thực hiện T1. Khi đó theo cơ chế của READ COMMITTED thì T2 chỉ được phép đọc dữ liệu đã commit mà T1 chưa commit nên T2 sẽ được trả lại kết quả ban đầu chưa update

### Tình huống 2: PHANTOM READ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ERR03: *Phantom Read***  T1 (User = cửa hàng (đối tác)): thực hiện thêm một chi nhánh mới  T2 (User = khách hàng): thực hiện xem danh sách đối tác | | | | |
| sp\_DT\_ThemChiNhanh |  | sp\_KH\_XemDSDoiTac |  |
| ***Input:*** @machinhanh  @diachi  @sdt  @macuahang  @mahopdong  @khuvuc  ***Output***: 1: thêm thành công  0: thêm thất bại | ***Input***:  ***Output***: : 1: cập nhật thành công  0: cập nhật thất bại |
| SET TRANSACTION ISOLATION  LEVEL …. | SET TRANSACTION ISOLATION  LEVEL SERIALIZEBLE |
|  |  | BEGIN TRAN |  |
|  |  | B1: Xem danh sách đối tác(cửa hàng)  SELECT MA\_CUA\_HANG, TEN\_CUA\_HANG, EMAIL, THANH\_PHO, QUAN, SDT, SO\_CHI\_NHANH  FROM CUAHANG | **S(CUAHANG)**  //Xin khóa đọc trên bảng CUAHANG  (xin thành công) |
|  |  | WAITFOR DELAY '00:00:10' |  |
| BEGIN TRAN |  |  |  |
| B1: Kiểm tra trùng lắp của @machinhanh và @diachi  IF(EXISTS(SELECT \* FROM CHINHANH WHERE MA\_CUA\_HANG = @macuahang AND DIA\_CHI = @diachi))  begin  rollback tran  RETURN -1  end    IF(EXISTS(SELECT \* FROM CHINHANH WHERE MA\_CHI\_NHANH = @machinhanh))  begin  rollback tran  RETURN -1  end | **S(CHI\_NHANH)**  //Xin khoá đọc trên bảng CHI\_NHANH với điều kiện  @machinhanh = ‘CN001’  (xin thành công)  **S(CHI\_NHANH)**  //Xin khoá đọc trên bảng CHI\_NHANH với điều kiện  @diachi = ‘Dĩ An, Bình Dương’  (xin thành công) |  |  |
| B2: Thêm chi nhánh  INSERT INTO CHINHANH(MA\_CHI\_NHANH, SL\_DON\_MOINGAY, DIA\_CHI, SDT, TINH\_TRANG, MA\_CUA\_HANG, MA\_HOP\_DONG, KHUVUC)  VALUES(@machinhanh,NULL,@diachi, @sdt, null, @macuahang, @mahopdong, @khuvuc) | **X(CHI\_NHANH)**  //Xin khóa ghi trên bảng CHINHANH  (xin thành công) |  |  |
| B3: Cập nhập số lượng chi nhánh  UPDATE CUAHANG  SET SO\_CHI\_NHANH = SO\_CHI\_NHANH + 1  WHERE MA\_CUA\_HANG = @macuahang | **UX(CUAHANG)**  //Xin khóa ghi trên bảng CUAHANG |  |  |
| COMMIT TRAN |  |  |  |
|  |  | B1: Xem danh sách đối tác(cửa hàng)  SELECT MA\_CUA\_HANG, TEN\_CUA\_HANG, EMAIL, THANH\_PHO, QUAN, SDT, SO\_CHI\_NHANH  FROM CUAHANG | **S(CUAHANG)**  //Xin khóa đọc trên bảng CUAHANG  (xin thành công) |
|  |  | COMMIT TRAN |  |

Giải thích: Để khắc phục lỗi Phantom Read, ở đây nhóm đề xuất đặt T2 ở mức cô lập Serializable, khi đó nó sẽ đặt 1 khóa ở T2 ngăn chặn các giao tác khác thêm hay cập nhập trong T2, qua đó giải quyết được Phantom Read. Khi thực hiện T2, T1 bị chặn cho đến khi T2 hoàn thành và kết thúc T2, cả 2 lần đọc điều nhận được giá trị như nhau

## **4. Sinh viên thực hiện: Nguyễn Nhật Trường**

### 

### Tình huống 1: UNREPEATABLE READ

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: UNREPEATABLE READ**  T1 (User = khách hàng): xem tình trạng đơn hàng  T2 (User = tài xế): tài xế thực hiện cập nhật tình trạng đơn hàng | | | |
| **USP\_KIEMTRA\_TTGH** | **Khóa** | **USP\_TX\_CAPNHAT\_TTGH** | **Khóa** |
| ***Input:*** @MA\_KHACH\_HANG CHAR(5), @MA\_DON CHAR(5) ***Output***: | ***Input***: @MA\_TAI\_XE CHAR(5), @MA\_DON CHAR(5), @TTGH NVARCHAR(255)  ***Output***: |
| BEGIN TRAN |  |  |  |
| **SET TRAN ISOLATION LEVEL REPEATABLE READ** | Thiết lập khóa S khi đọc và giữ đến khi hết giao tác |  |  |
| IF NOT EXISTS(SELECT DDH.\* FROM DONDATHANG DDH WHERE DDH.MA\_KHACH\_HANG=@MA\_KHACH\_HANG AND DDH.MA\_DON=@MA\_DON)  BEGIN  PRINT N'ĐH không tồn tại'  RETURN  END | **S(DONDATHANG)**  T1 xin khóa đọc trên bảng DONDATHANG. T1 sẽ nhả khóa sau khi commit. |  |  |
| DECLARE @TTGH NVARCHAR(255)  SET @TTGH = 0  SET @TTGH = (SELECT TINH\_TRANG FROM DONDATHANG WHERE MA\_DON = @MA\_DON) | **S(DONDATHANG)** T1 xin khóa đọc trên bảng DONDATHANG. T1 sẽ nhả khóa sau khi commit. |  |  |
| WAITFOR DELAY '00:00:015' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | UPDATE DONDATHANG  SET TINH\_TRANG = @TTGH  WHERE MA\_DON = @MA\_DON AND MA\_TAI\_XE = @MA\_TAI\_XE | **U,X(DONDATHANG)**  T1 xin khóa ghi trên bảng DONDATHANG thành công. T2 sẽ chờ T1 commit để xin khóa. |
|  |  | COMMIT TRAN |  |
| IF ((SELECT TINH\_TRANG FROM DONDATHANG WHERE MA\_DON = @MA\_DON) <> @TTGH)  BEGIN  RAISERROR(N'Dữ liệu hai lần đọc khác nhau',16,1)  END  SELECT \*  FROM DONDATHANG  WHERE MA\_DON = @MA\_DON | **S(DONDATHANG)**  T1 xin khóa ghi trên bảng DONGIAOHANG thành công. T2 sẽ chờ T1 commit để xin khóa. |  |  |
| COMMIT TRAN |  |  |  |

**GIẢI THÍCH:** Áp dụng mức cô lập REPEATABLE READ đối với T1, làm cho T2 không thể chen ngang UPDATE dữ liệu khi T1 đang thực hiện. Kết quả hai lần đọc cho ra TINHTRANG đều cho kết quả giống nhau.

### Tình huống 2: Dirty Read

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: DIRTY READ**  T1 (User = Đối tác (cửa hàng)): Đối tác thực hiện cập nhật giá của món ăn B (@MA\_MON\_AN = ‘MA002’)  T2 (User = Khách hàng): Khách hàng xem giá món ăn B ((@MA\_MON\_AN = ‘MA002’) | | | |
| **USP\_DT\_CAPNHAT\_GIA\_MONAN** | **Khóa** | **USP\_KH\_XEMGIA\_MONAN** | **Khóa** |
| ***Input:*** @MA\_MON\_AN CHAR(5), @GIA\_UPDATE MONEY  ***Output***: ……. | ***Input***: @MA\_MON\_AN CHAR(5)  ***Output***: |
| BEGIN TRAN |  |  |  |
| B1: kiểm tra @MA\_MON\_AN tồn tại    IF NOT EXISTS (SELECT \* FROM MON\_AN WHERE MA\_MON\_AN=@MA\_MON\_AN)  BEGIN  PRINT N'MÓN ĂN KHÔNG TỒN TẠI'  RETURN  END | **S(MON\_AN)**  T1 xin khóa đọc trên bảng MON\_AN ứng với MA\_MON\_AN=@MA\_MON\_AN |  |  |
| DECLARE @GIASP MONEY  SET @GIASP = (SELECT DON\_GIA FROM MON\_AN WHERE MA\_MON\_AN=@MA\_MON\_AN)  UPDATE MON\_AN  SET DON\_GIA = @GIA\_UPDATE  WHERE MA\_MON\_AN=@MA\_MON\_AN | **S(MON\_AN)**  T1 xin khóa đọc trên bảng MON\_AN ứng với MA\_MON\_AN=@MA\_MON\_AN |  |  |
| WAITFOR DELAY '00:00:05' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | SET TRAN ISOLATION LEVEL READ COMMITTED |  |
|  |  | SELECT \*  FROM MON\_AN WITH(NOLOCK)  WHERE MA\_MON\_AN=@MA\_MON\_AN | **S(MON\_AN)**  T2 xin khóa đọc trên bảng MON\_AN ứng với MA\_MON\_AN=@MA\_MON\_AN |
|  |  | COMMIT TRAN |  |
| IF (@GIA\_UPDATE = 0)  BEGIN  ROLLBACK TRAN  RETURN  END |  |  |  |
| COMMIT TRAN |  |  |  |
|  |  |  |  |

GIẢI THÍCH: Việc áp dụng NOLOCK trên bảng MON\_AN để cho kết quả nhanh chóng đã dẫn đến tình trạng DIRTY READ, giao tác T2 đọc dữ liệu chưa được commit bởi T1. Để tránh tình trạng Dirty Read, nhóm đề xuất xóa NOLOCK đối với bảng MON\_AN, để đưa T2 về mức cô lập READ COMMITTED. Lúc này, quá trình thực hiện của hai giao tác trên được thực hiện dựa trên mũi tên màu đỏ ở trên. Đầu tiên, T1 thực hiện select và update giá trên bảng MON\_AN, cùng lúc đó, T2 thực hiện đọc giá sản phẩm của T1 vừa cập nhật. Tiếp đến, vì giá sản phẩm bị nhập sai, nên T1 đã rollback, theo cơ chế của mức cô lập READ COMMITTED, thì T2 chỉ được phép đọc dữ liệu đã commit, vì thế kết quả đọc dữ liệu của T2 sẽ được trả lại kết quả ban đầu chưa thực hiện update.

### Tình huống 3: Circle Deadlock

|  |  |  |  |
| --- | --- | --- | --- |
| **ERR01: CIRCLE DEADLOCK**  T1: Người quản trị 1 cập nhật thông tin tài khoản của khách hàng (đổi mật khẩu) và cập nhật thông tin của tài xế (Email).  T2: Quản trị 2 cập nhật thông tin của tài xế(Email) và cập nhật thông tin tài khoản của khách hàng (đổi mật khẩu). | | | |
| **USP\_QT1\_CAPNHAT\_TT @TEN\_DANG\_NHAP** | **Khóa** | **USP\_QT2\_CAPNHAT\_TT** | **Khóa** |
| ***Input:*** @TEN\_DANG\_NHAP NVARCHAR(255), @MAT\_KHAU\_MOI CHAR(100), @MATX CHAR(5), @EMAIL\_MOI CHAR(100)  ***Output***: ……. | ***Input***: @TEN\_DANG\_NHAP NVARCHAR(255), @MAT\_KHAU\_MOI CHAR(100), @MATX CHAR(5), @EMAIL\_MOI CHAR(100)  ***Output***: |
| BEGIN TRAN |  |  |  |
| B1: kiểm tra @TEN\_DANG\_NHAP tồn tại    IF NOT EXISTS(SELECT TK.\* FROM TAIKHOAN TK WHERE TEN\_DANG\_NHAP=@TEN\_DANG\_NHAP)  BEGIN  PRINT N'Tài khoản không tồn tại'  RETURN  END | **S(TAIKHOAN)**  T1 xin khóa ghi trên bảng TAIKHOAN. |  |  |
| UPDATE TAIKHOAN SET MAT\_KHAU = @MAT\_KHAU\_MOI WHERE TEN\_DANG\_NHAP=@TEN\_DANG\_NHAP | **U,X(TAIKHOAN)**  T1 xin khóa trên bảng TAIKHOAN |  |  |
| WAITFOR DELAY '00:00:05' |  |  |  |
|  |  | BEGIN TRAN |  |
|  |  | UPDATE TAIKHOAN SET MAT\_KHAU = @MAT\_KHAU\_MOI WHERE TEN\_DANG\_NHAP=@TEN\_DANG\_NHAP | **U,X(TAIXE)**  T2 xin khóa trên bảng TAIKHOAN. T2 chờ T1 commit thì bắt đầu thực hiện. |
|  |  | WAITFOR DELAY '00:00:05' |  |
| UPDATE TAIXE  SET EMAIL = @EMAIL\_MOI  WHERE MA\_TAI\_XE = @MATX | **U,X(TAIXE)**  T1 xin khóa trên bảng TAIKHOAN |  |  |
|  |  | UPDATE TAIXE  SET EMAIL = @EMAIL\_MOI  WHERE MA\_TAI\_XE = @MATX | **S(TAIKHOAN)**  T1 xin khóa trên bảng TAIKHOAN |
| COMMIT TRAN |  |  |  |
|  |  | COMMIT TRAN |  |

GIẢI THÍCH:Về lỗi Circle Deadlock không có cách nào để xử lý tranh chấp theo các cách thông thường nên em đề xuất việc thay đổi thứ tự thao tác đối với T2, để tránh tình trạng gây ra CYCLE DEADLOCK. Khi đó, vì cả 2 giao tác ở mức cô lập COMMITTED cho nên T1 thực hiện thực hiện và commit thì T2 sẽ bắt đầu thực hiện. → Cả 2 giao tác đều được thực thi thành công