**ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**Khoa Công Nghệ Thông Tin**

**BÁO CÁO CUỐI KỲ**

**RESTAURANT SYSTEM**

**Giảng viên hướng dẫn:**

***Hoàng Anh Tú***

**Các thành viên trong nhóm 40:**

***Nguyễn Bá Ngọc – MSSV : 1141085***

***Bùi Trung Tân – MSSV :1141129***

***Nguyễn Minh Nghĩa – MSSV : 1141083***

***Nguyễn Minh Vũ – MSSV : 1141187***

## Phụ Lục

[1. Thông tin nhóm 1](#_Toc325069036)

[1.1 Thông tin thành viên: 1](#_Toc325069037)

[2. Các tranh chấp đồng thời đã phát hiện 1](#_Toc325069038)

[2.1 1141085 – Nguyễn Bá Ngọc 1](#_Toc325069039)

[2.1.1 Các trường hợp tranh chấp 1](#_Toc325069040)

[2.1.2 Xử lý các tranh chấp 2](#_Toc325069041)

[2.2 1141187 – Nguyễn Minh Vũ 2](#_Toc325069042)

[2.2.1 Các trường hợp tranh chấp 2](#_Toc325069043)

[2.2.2 Xử lý các tranh chấp 3](#_Toc325069044)

[2.3 1141083 – Nguyễn Minh Nghĩa 3](#_Toc325069045)

[2.3.1 Các trường hợp tranh chấp 3](#_Toc325069046)

[2.3.2 Xử lý các tranh chấp 3](#_Toc325069047)

[2.4 1141129 – Bùi Trung Tân 4](#_Toc325069048)

[2.4.1 Các trường hợp tranh chấp 4](#_Toc325069049)

[2.4.2 Xử lý các tranh chấp 4](#_Toc325069050)

# Thông tin nhóm

## Thông tin thành viên:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **MSSV** | **Họ tên** | **Nhóm trưởng** | **Đánh giá** |
| 1 | 1141085 | Nguyễn Bá Ngọc | X | 100% |
| 2 | 1141187 | Nguyễn Minh Vũ |  |  |
| 3 | 1141083 | Nguyễn Minh Nghĩa |  |  |
| 4 | 1141129 | Bùi Trung Tân |  |  |

# Các tranh chấp đồng thời đã phát hiện

## 1141085 – Nguyễn Bá Ngọc

### Các trường hợp tranh chấp

#### Drity read

#### Lost update

#### Unrepeatable tread

#### Phantom

#### deadlock

### Xử lý các tranh chấp

#### Drity read

#### Lost update

#### Unrepeatable tread

#### Phantom

#### deadlock

## 1141187 – Nguyễn Minh Vũ

### Các trường hợp tranh chấp

#### Drity read

#### Lost update

#### Unrepeatable tread

#### Phantom

#### deadlock

### Xử lý các tranh chấp

#### Drity read

#### Lost update

#### Unrepeatable tread

#### Phantom

#### deadlock

## 1141083 – Nguyễn Minh Nghĩa

### Các trường hợp tranh chấp

#### Drity read

#### Lost update

#### Unrepeatable tread

#### Phantom

#### deadlock

### Xử lý các tranh chấp

#### Drity read

#### Lost update

#### Unrepeatable tread

#### Phantom

#### deadlock

## 1141129 – Bùi Trung Tân

### Các trường hợp tranh chấp

#### Drity read

###### **Kịch bản :**

|  |  |
| --- | --- |
| T1: thực hiện chuyển bàn ăn A sang B | T2: ghép bàn |
| Cập nhật trạng thái bàn A thành ‘trống’ |  |
|  | Lấy danh sách bàn trống để chuyển bàn.  (hoặc ghép bàn, xếp bàn…) |
| Kiểm tra bàn B có đang trống hay không:   * Nếu không còn trống (do bị giao tác khác đã thay đổi trạng thái bàn ăn này)   =>Rollback: trả lại trạng thái ban ‘đang sử dụng’ cho bàn A   * Ngược lại thì cập nhật trạng thái bàn B thành ‘đang sử dụng’.   =>Hoàn tất nghiệp vụ |  |

* Tại thời điểm T2 đã đọc dữ liệu, được trạng thái bàn A và B đều là ‘trống’ và được đọc lên.
  + Nếu T1 bị rollback, trạng thái bàn A bị gán lại thành ‘đang sử dụng’, T2 đã đọc sai dữ liệu khi đọc được bàn A.
  + Nếu T1 thành công, trạng thái bàn B bị gán thành ‘đang sử dụng’, vậy T2 đã đọc sai dữ liệu khi đọc được bàn B.

###### **Stored Procedure :**

|  |  |
| --- | --- |
| **T1** | **T2** |
| create proc sp\_ChuyenBan @kq int out, @mabanOLD int, @mabanNEW int, @mahd int  as  begin  begin tran t1  SET @kq = 1    UPDATE BanAn SET TrangThai = 0  WHERE MaBan = @mabanOLD    waitfor delay '0:0:10' |  |
|  | create proc sp\_selectBanAn @trangthai int, @manh nchar(10)  as  begin tran  IF (@trangthai = -1)  SELECT \* FROM BanAn  WHERE MaNH = @manh  ELSE  SELECT \* FROM BanAn  WHERE TrangThai = @trangthai  AND MaNH = @manh  commit tran |
| IF( (SELECT TrangThai FROM BanAn  WHERE MaBan = @mabanNEW) = 0)  begin  EXEC dbo.sp\_UpdateTrangThaiBanAn @mabanNEW, 1  UPDATE HoaDon SET MaBan = @mabanNEW WHERE MaHD = @mahd  commit tran t1  end  ELSE  begin  SET @kq = 0  rollback tran t1  end  end |  |

###### **Nguyên nhân tranh chấp :**

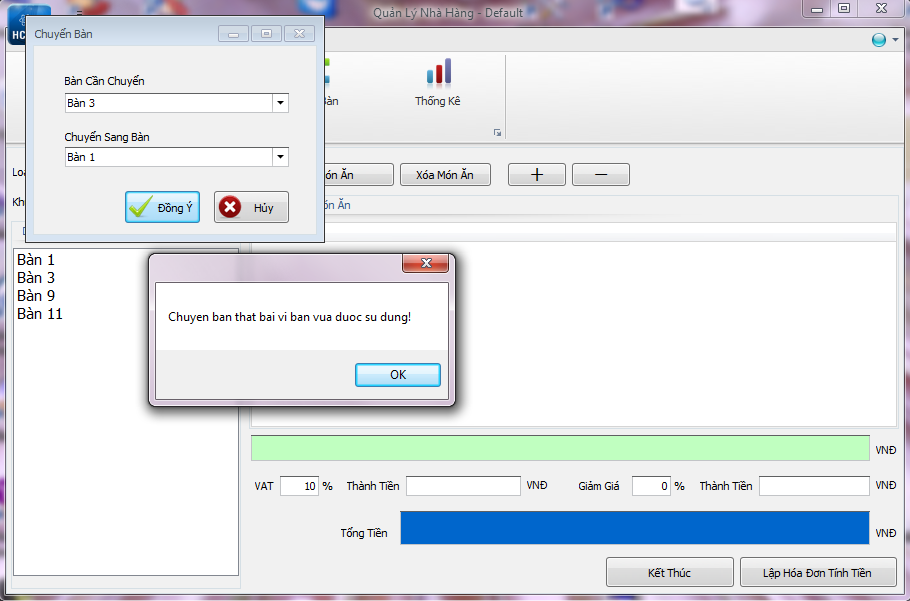
Với mức cô lập Read Uncommited, T2 đọc mà không xin khoá đọc. Như vậy bàn ăn A đang có khoá ghi (T1 update trạng thái) và T2 vẫn có thể đọc được dữ liệu, nên T2 có thể bị đọc sai.

###### **Cách khắc phục:**

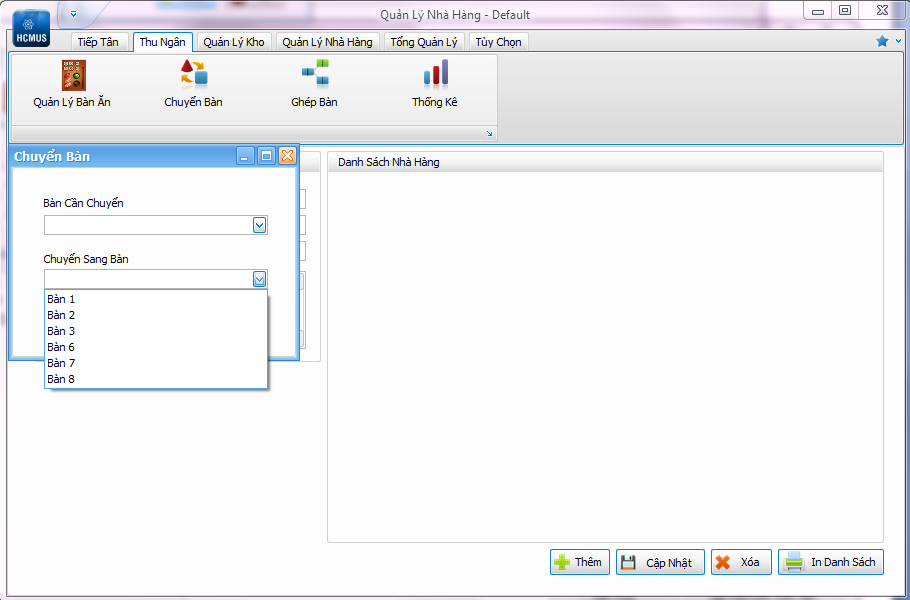
Thiết lập mức cô lập Read Commited cho T1 và T2. T2 phải xin khoá đọc trên bàn A và bàn B, như vậy T2 phải đợi T1 hoàn tất.

###### **Giao diện chương trình bị Dirty Read:**

Giao tác T1



Giao tác T2



Xuất hiện dữ liệu rác:

Bàn số 3

#### Lost update

###### **Kịch bản :**

|  |  |
| --- | --- |
| T1: thực hiện cộng thêm 1 vào số lượng của món ăn A trong chi tiết hoá đơn rồi hiển thị ra giao diện | T2: thực hiện trừ đi 1 vào số lượng của món ăn A trong chi tiết hoá đơn rồi hiển thị ra giao diện |
| Đọc số lượng hiện tại (n) của món ăn A vào biến  Cộng 1 vào biến |  |
|  | Đọc số lượng hiện tại (n) của món ăn A vào biến  Cộng 1 vào biến |
| Cập nhật lại giá trị của biến vào số lượng của món A  Số lượng mới: n + 1 |  |
| (Tại đây có một giao tác T3 đọc ra số lượng của món A) |  |
|  | Cập nhật lại giá trị của biến vào số lượng của món A  Số lượng mới: n - 1 |

* Như vậy sau khi T1 và T2 thực hiện xong, số lượng cúa món A là n – 1. Trong khi đúng ra số lượng món A phải là n.
  + Dữ liệu n + 1 đã bị ghi đè bởi n – 1
  + T3 bị dirty read vì ngay sau khi đọc xong thì giá trị vừa đọc bị thay đổi.

###### **Stored Procedure :**

|  |  |
| --- | --- |
| **T1** | **T2** |
| create proc sp\_UpdateSoLuongCTHD\_1 @mahd int, @mamon int, @soluong int  as  begin tran  SET TRAN ISOLATION LEVEL  READ UNCOMMITTED    declare @soluong\_hientai int  SELECT @soluong\_hientai = SoLuong  FROM ChiTietHoaDon  WHERE MaHD = @mahd  AND MaMon = @mamon  SET @soluong\_hientai = @soluong\_hientai + @soluong    waitfor delay '0:0:7' |  |
|  | create proc sp\_UpdateSoLuongCTHD\_1 @mahd int, @mamon int, @soluong int  as  begin tran  SET TRAN ISOLATION LEVEL  READ UNCOMMITTED    declare @soluong\_hientai int  SELECT @soluong\_hientai = SoLuong  FROM ChiTietHoaDon  WHERE MaHD = @mahd  AND MaMon = @mamon  SET @soluong\_hientai = @soluong\_hientai + @soluong  waitfor delay '0:0:7' |
| UPDATE ChiTietHoaDon SET SoLuong = @soluong\_hientai WHERE MaHD = @mahd AND MaMon = @mamon  commit tran |  |
|  | UPDATE ChiTietHoaDon SET SoLuong = @soluong\_hientai WHERE MaHD = @mahd AND MaMon = @mamon  commit tran |

###### **Nguyên nhân tranh chấp :**

Sau khi T1 đọc dữ liệu, T2 cũng có thể đọc dữ liệu (Shared-lock không có tranh chấp khi cấp trên cùng một đơn vị dữ liệu). Như vậy cả T1 và T2 đều đọc được trạng thái nguyên thuỷ của dữ liệu, do đó khi update vào dữ liệu đã đọc được thì đơn vị dữ liệu chỉ nhận được giá trì từ T1 hoặc T2.

T3 đọc dữ liệu ngay sau khi T1 update mà không biết rằng T2 vẫn còn update lên cùng đơn vị dữ liệu đó, nên T3 bị dirty read.

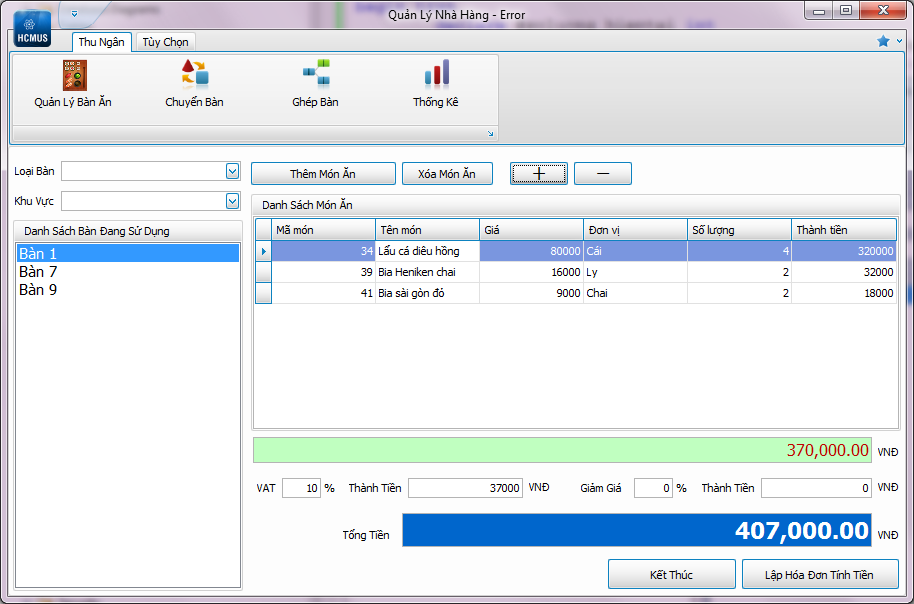
###### **Cách khắc phục:**

Nâng cấp khoá đọc thành Uplock, như vậy T2 phải đợi T1 hoàn tất thì mới có thể đọc và cập nhất giá trị. Theo kịch bản trên thì T3 bắt đầu sau T2, thiết lập mức cô lập Read Committed cho T3 thì T3 phải đợi T2 hoàn tất mới có thể đọc giá trị và không bị Dirty Read.

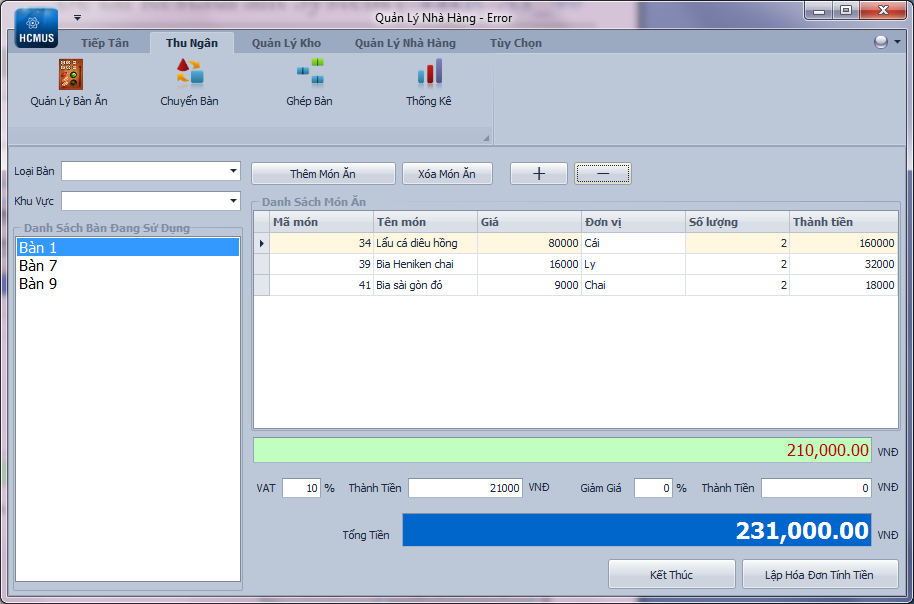
###### **Giao diện chương trình bị Lost Update:**

Số lượng của mã món 34 ban đầu bằng 3. T1 chạy trước tăng 1, T2 chạy sau giảm 1. Như vậy số lượng của món 34 sẽ là 2 (bị lost update) nhưng trên giao diện T1 lại hiển thị là 4 do T3 đọc được số lượng của món 34 ngay sau khi T1 ghi 4.

Giao tác T1



Giao tác T2



#### Unrepeatable tread

###### **Kịch bản :**

|  |  |
| --- | --- |
| T1: thực hiện thêm một món ăn A vào hoá đơn | T2: xoá món ăn A |
| Kiểm tra xem món ăn A có tồn tại hay không (đọc lần 1) |  |
|  | Xoá món ăn A |
| Nếu món ăn A có tồn tại (kiến quả của đọc lần 1) thì   * Đọc lần 2 để lấy ra giá của món ăn A * Thêm thông tin món ăn vào bảng chi tiết hoá đơn |  |

* Tại thời điểm T1 kiểm tra xem món ăn có tồn tại hay không (đọc lần 1).
  + Giả sử lần đọc đầu tiên trả ra kết quả là có, các câu lệnh bên trong hàm IF được thực hiện
  + Trước khi T1 đọc lần thứ 2, T2 đã xoá món ăn đó, làm cho T1 không thể đọc lại món ăn đó để lấy ra Giá của món ăn.

###### **Stored Procedure :**

|  |  |
| --- | --- |
| T1 | T2 |
| create proc sp\_insertCTHD @mahd int, @mamon int, @soluong int  as  begin tran  IF( (select COUNT(\*) from MonAn where MaMon = @mamon) = 1 )  begin  waitfor delay '0:0:10' |  |
|  | DELETE MonAn WHERE MaMon = @mamon |
| declare @dongia float  SET @dongia = 0  set @dongia = (select Gia from MonAn where MaMon = @mamon)  INSERT INTO ChiTietHoaDon(MaHD, MaMon, SoLuong, ThanhTien) VALUES(@mahd, @mamon, @soluong, @dongia \* @soluong)  end  commit tran |  |

###### **Nguyên nhân tranh chấp:**

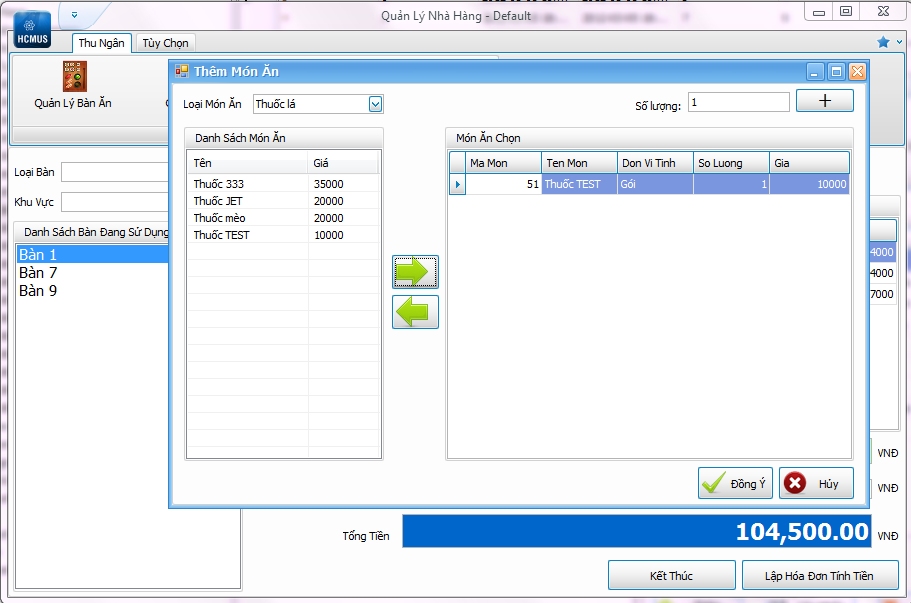
Khi T1 được thiết lập mức cô lập là Read Uncommitted hoặc Read Commited, sau lần đọc thứ nhất, khoá đọc được giải phóng trên món ăn A ngay sau khi đọc xong. Như vậy giao tác T2 có thể thay đổi thông tin món ăn A (xoá, thay đổi điều kiện lọc), làm cho T1 không thể đọc lại món ăn A ở lần đọc thứ hai.

###### **Cách khắc phục:**

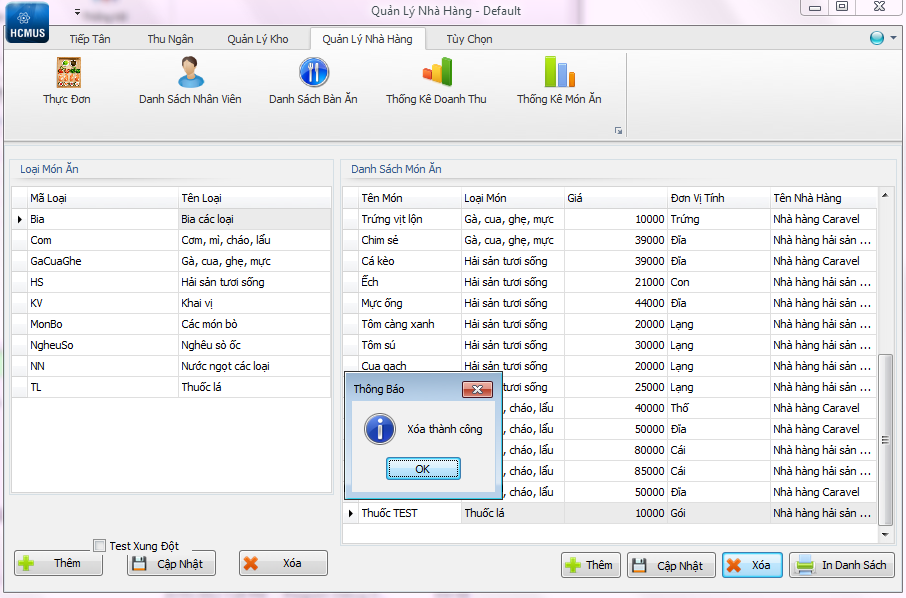
Thiết lập mức cô lập Repeatable Read cho T1. Như vậy ở lần đọc thứ nhất, khoá đọc được giữ trên món ăn A đến khi T1 kết thúc, T2 phải đợi T1 hoàn tất thì mới có thể thực hiện câu lệnh xoá món ăn.

###### **Giao diện chương trình bị Unrepeatable Read:**

Giao tác T1



Giao tác T2



#### Phantom

#### Deadlock

###### **Kịch bản :**

|  |  |
| --- | --- |
| T1: cập nhật số lượng cho chi tiết hoá đơn | T2: cập nhật số lượng cho chi tiết hoá đơn |
| Kiểm tra xem món ăn A cần cập nhật số lượng đã có trong chi tiết hoá đơn chưa. |  |
|  | Kiểm tra xem món ăn A cần cập nhật số lượng đã có trong chi tiết hoá đơn chưa. |
| Nếu có rồi thì cập nhật lại số lượng cho món ăn A |  |
|  | Nếu có rồi thì cập nhật lại số lượng cho món ăn A |

* Để không bị Unrepeatable Read, mức cô lập được sử dụng ở T1 và T2 là Repeatable Read. Như vậy xuất hiện Conversion Deadlock khi shared lock được cấp và giữ đến cuối giao tác.

###### **Stored Procedure:**

|  |  |
| --- | --- |
| T1 | T2 |
| create proc sp\_UpdateSoLuongCTHD @mahd int, @mamon int, @soluong int  as  begin tran  SET TRAN ISOLATION LEVEL  REPEATABLE READ    IF( (SELECT COUNT(\*) FROM ChiTietHoaDon  WHERE MaHD = @mahd  AND MaMon = @mamon) = 1 )  begin  waitfor delay '0:0:7' |  |
|  | create proc sp\_UpdateSoLuongCTHD @mahd int, @mamon int, @soluong int  as  begin tran  SET TRAN ISOLATION LEVEL  REPEATABLE READ    IF( (SELECT COUNT(\*) FROM ChiTietHoaDon  WHERE MaHD = @mahd  AND MaMon = @mamon) = 1 )  begin  waitfor delay '0:0:7' |
| UPDATE ChiTietHoaDon SET SoLuong = @soluong WHERE MaHD = @mahd AND MaMon = @mamon  End  commit tran |  |
|  | UPDATE ChiTietHoaDon SET SoLuong = @soluong WHERE MaHD = @mahd AND MaMon = @mamon  **E**nd  commit tran |

###### **Nguyên nhân tranh chấp:**

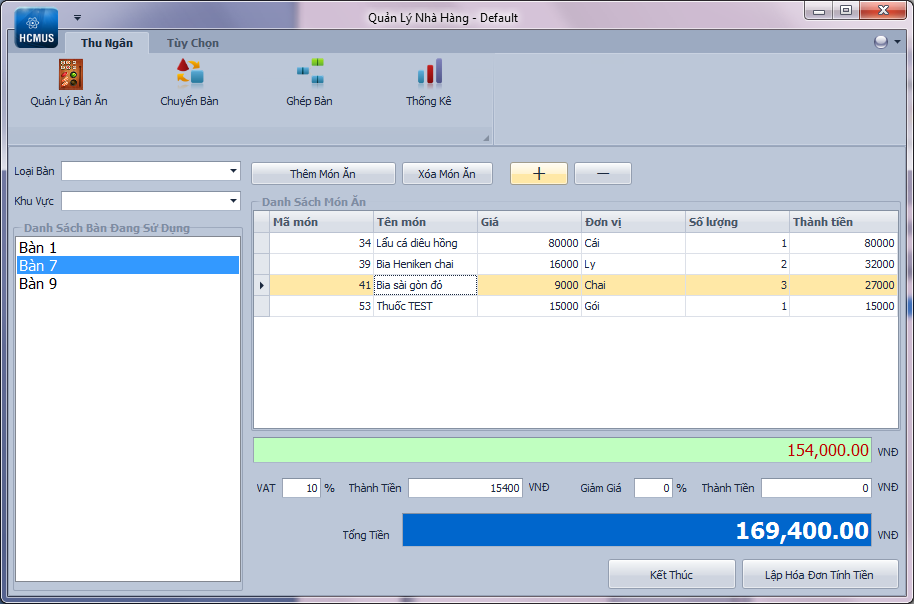
* Để cập nhật số lượng món ăn trong chi tiết hoá đơn, cần đảm bảo trong chi tiết hoá đơn có tồn tại món ăn đó, nên cần thực hiện kiểm tra
* Để đảm bảo sau khi kiểm tra, món ăn không bị thay đổi thông tin (ví dụ: xoá) thì cần thiết lập mức cô lập Repeatable Read. Như vậy ở khoá đọc được cấp và giữ trên món ăn A đến hết giao tác.
* T1 cấp shared lock trên món ăn A và giữ đến hết giao tác. T2 cấp shared lock trên món ăn A và giữ đến hết giao tác. T1 cần cập nhật giữ liệu trên món ăn A, nên phải chờ T2 kết thúc. T2 cũng cần cập nhật giữ liệu trên món ăn A và phải chờ T1 kết thúc.
* Xuất hiện Conversion Deadlock.

###### **Cách khắc phục:**

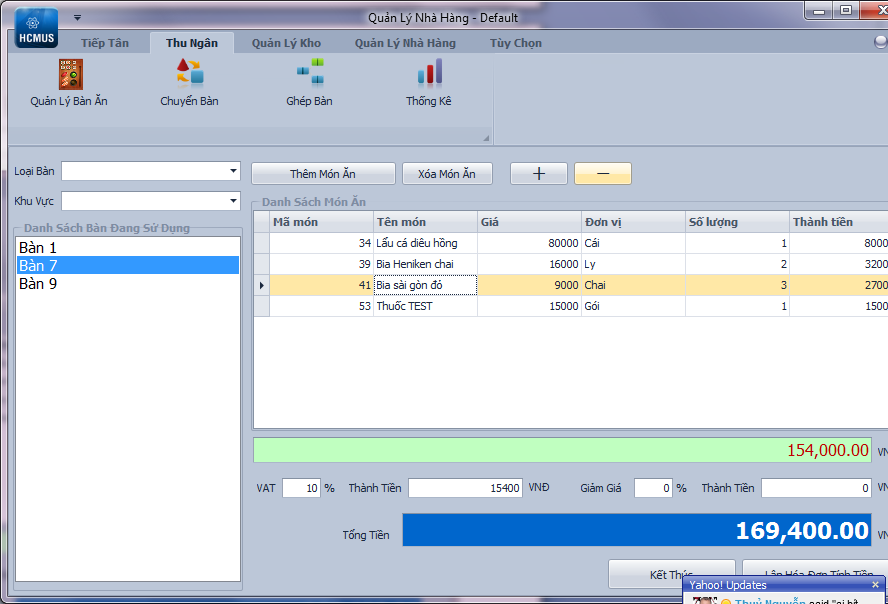
Thiết lập timeout (SET LOCK\_TIMEOUT 20000) để chỉ có một giao tác được thực thi, giao tác còn lại bị huỷ.

###### **Giao diện chương trình bị Conversion Deadlock:**

Giao tác T1: (tăng số lượng món ăn lên 1)



Giao tác T2: (giảm số lượng món ăn đi 1)



### Xử lý các tranh chấp

#### Drity read

#### Lost update

#### Unrepeatable tread

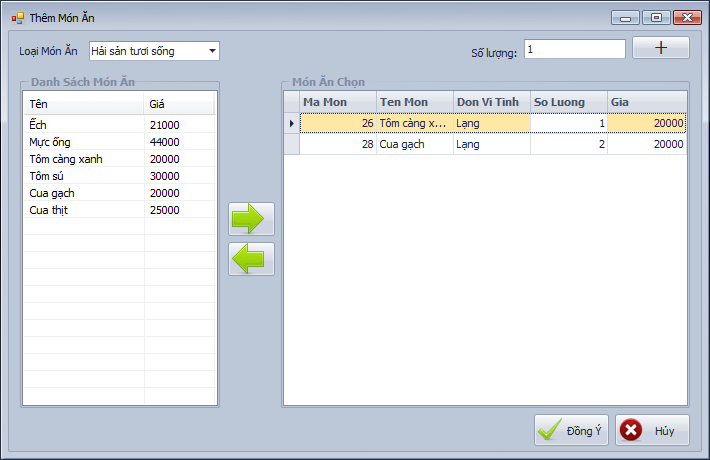
#### Phantom

#### deadlock

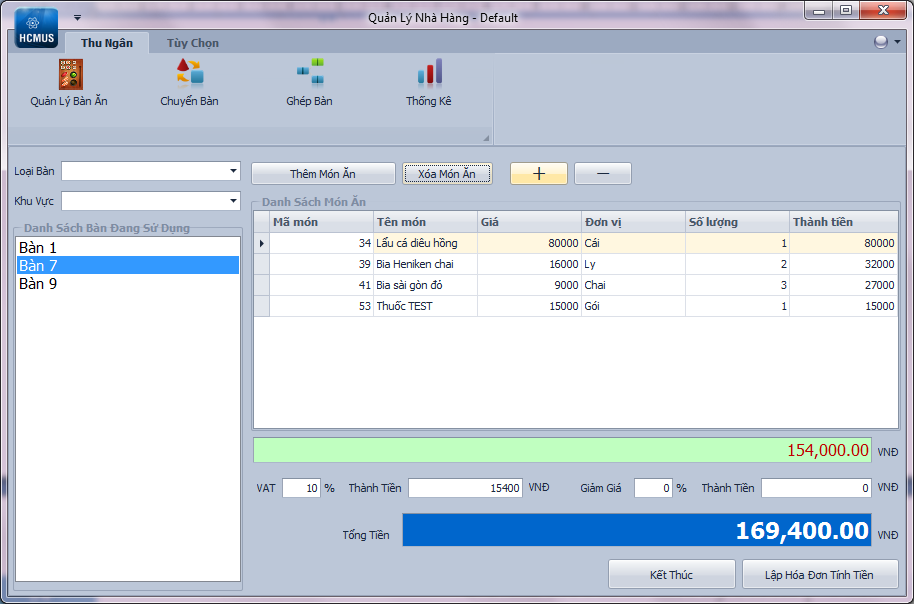
1. **Các chức năng trong đề tài**
   1. **Nhân Viên Tiếp Tân**
   2. **Nhân Viên Thu Ngân**

Đăng nhập với loại tài khoản là Thu ngân hoặc Quản lý nhà hàng, sử dụng Ribbon Thu Ngân

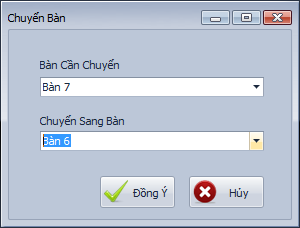
* + 1. **Thêm món ăn**

****

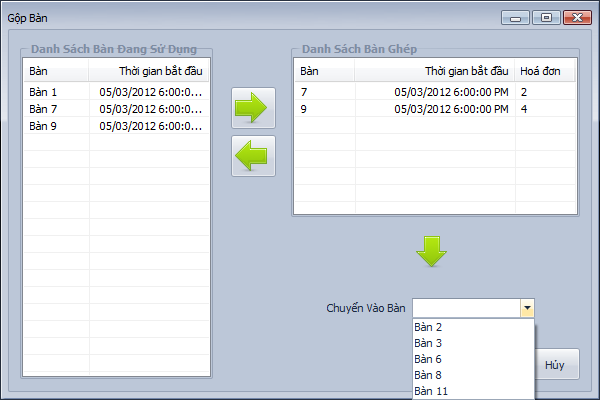
* + 1. **Xoá món ăn, cập nhật số lượng món ăn**

****

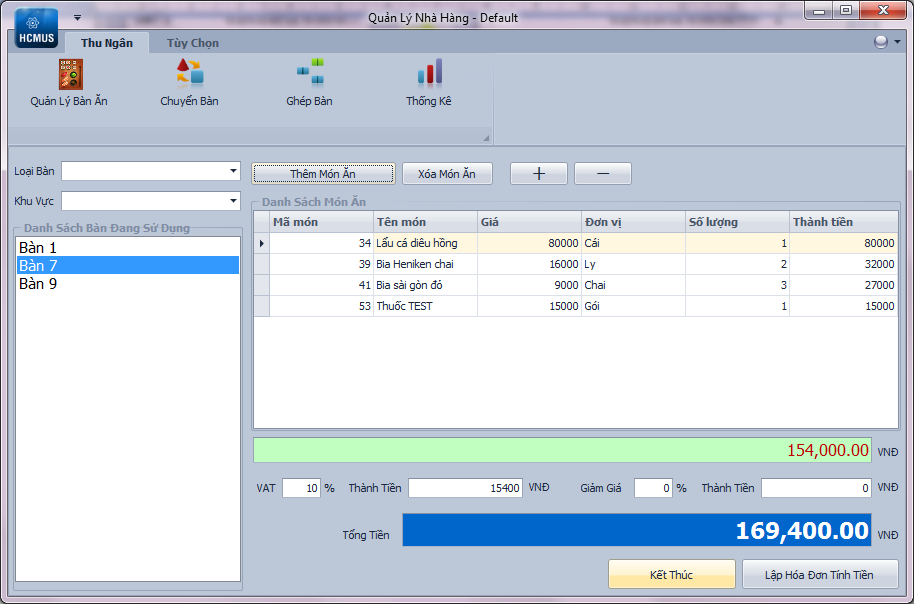
* + 1. **Chuyển bàn**

****

* + 1. **Ghép bàn**

****

* + 1. **Kết thúc sử dụng bàn**

****

* 1. **Nhân Viên Quản Lý Kho**
  2. **Quản Lý Nhà Hàng**