



TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN TPHCM

Khoa: Công nghệ thông tin

Bộ Môn: Hệ quản trị cơ sở dữ liệu

BÁO CÁO

ĐỒ ÁN THỰC HÀNH 2

- Nhóm thực hiện
 - Nhóm 8

- Giảng viên hướng dẫn
 - Tiết Gia Hồng

BẢNG THÔNG TIN THÀNH VIÊN

Nhóm: 8

Họ và tên	MSSV	Email
Nguyễn Thị Ngọc Hân	1712415	1712415@student.hcmus.edu.vn
Lai Gia Phú	1712662	giaphu260799@gmail.com
Nguyễn Đoàn Tấn Phúc	1712671	nguyendoantanhphuc@gmail.com
Trịnh Đức Thanh	1712769	1712769@student.hcmus.edu.vn
Dương Khánh Vi (Nhóm trưởng)	1712899	khanhvi696@gmail.com

BẢNG PHÂN CÔNG CÔNG VIỆC VÀ ĐÁNH GIÁ

Người thực hiện	Công việc thực hiện	Mức độ hoàn thành
1712415 – Nguyễn Thị Ngọc Hân	Phát sinh dữ liệu cho lược đồ.	100%
	Xóa món ăn trong đơn hàng.	
	Thêm/ xóa món ăn trong giỏ hàng.	
	Hiển thị các đơn hàng của thành viên.	
	Thêm đơn mới với dữ liệu từ giỏ hàng.	
	Hiển thị danh sách trạng thái các đơn hàng.	
	Thêm/ xóa một món ăn trong thực đơn.	
1712662 –	Xem thực đơn của một chi nhánh.	100%

Lai Gia Phú	Tỉ lệ hủy đơn theo các kênh đặt hàng/ loại khách hàng của 1 chi nhánh.	
	Tỉ lệ hủy đơn theo kênh đặt hàng/ loại khách hàng của toàn hệ thống.	
	Thêm/ xóa/ sửa món ăn.	
	Thêm/xóa một loại món ăn.	
	Code winform (chức năng của quản lí khách hàng và khách hàng).	
1712671 – Nguyễn Đoàn Tân Phúc	Cập nhật số lượng của món ăn.	100%
	Thống kê doanh thu theo ngày/ tháng/ năm của một chi nhánh.	
	Thống kê doanh thu theo món/ loại món của một chi nhánh.	
	Thống kê doanh thu theo kênh đặt hàng/ loại khách hàng của một chi nhánh.	
	Thống kê doanh thu theo ngày/ tháng/ năm của toàn hệ thống.	
	Thống kê doanh thu theo món/ loại món của toàn hệ thống.	
	Thống kê doanh thu theo kênh đặt hàng/ loại khách hàng của toàn hệ thống.	
1712769 – Trịnh Đức Thanh	Code winform (chức năng của quản lí chi nhánh và quản lí chung).	100%
	Thêm/ hủy đơn hàng.	
	Cập nhật tình trạng của đơn hàng.	

	<p>Khách hàng xem lại giỏ hàng.</p> <p>Thêm/ cập nhật món ăn vào đơn hàng.</p> <p>Thêm đơn hàng trực tuyến.</p> <p>Cập nhật nhân viên giao hàng của đơn hàng trực tuyến.</p> <p>Đánh báo cáo</p>	
1712899 – Dương Khánh Vi	<p>Tạo lược đồ cơ sở dữ liệu.</p> <p>Khách hàng đăng ký thẻ thành viên.</p> <p>Thành viên đăng nhập vào hệ thống.</p> <p>Nhân viên làm thẻ thành viên cho khách hàng.</p> <p>Thành viên cập nhật thông tin của mình</p> <p>Xóa thành viên.</p> <p>Phân công công việc.</p>	100%



MỤC LỤC

A. YÊU CẦU.....	1
B. KẾT QUẢ	1
I. Thiết kế cơ sở dữ liệu	1
1. Lược đồ cơ sở dữ liệu	1
2. Mô tả cơ sở dữ liệu.....	1
II. Đặc tả chức năng	6
III. Mô tả các stored procedure	7
IV. Mô tả kịch bản lỗi tranh chấp đồng thời/ deadlock	13
1. Mô tả thủ tục tranh chấp.....	13
2. Chi tiết các thủ tục tranh chấp.....	19
2.1. Dirty Read	19
2.2. Unrepeatable Read	34
2.3. Phantom.....	50
2.4. Lost Update	66
2.5. Deadlock	85

A. YÊU CẦU

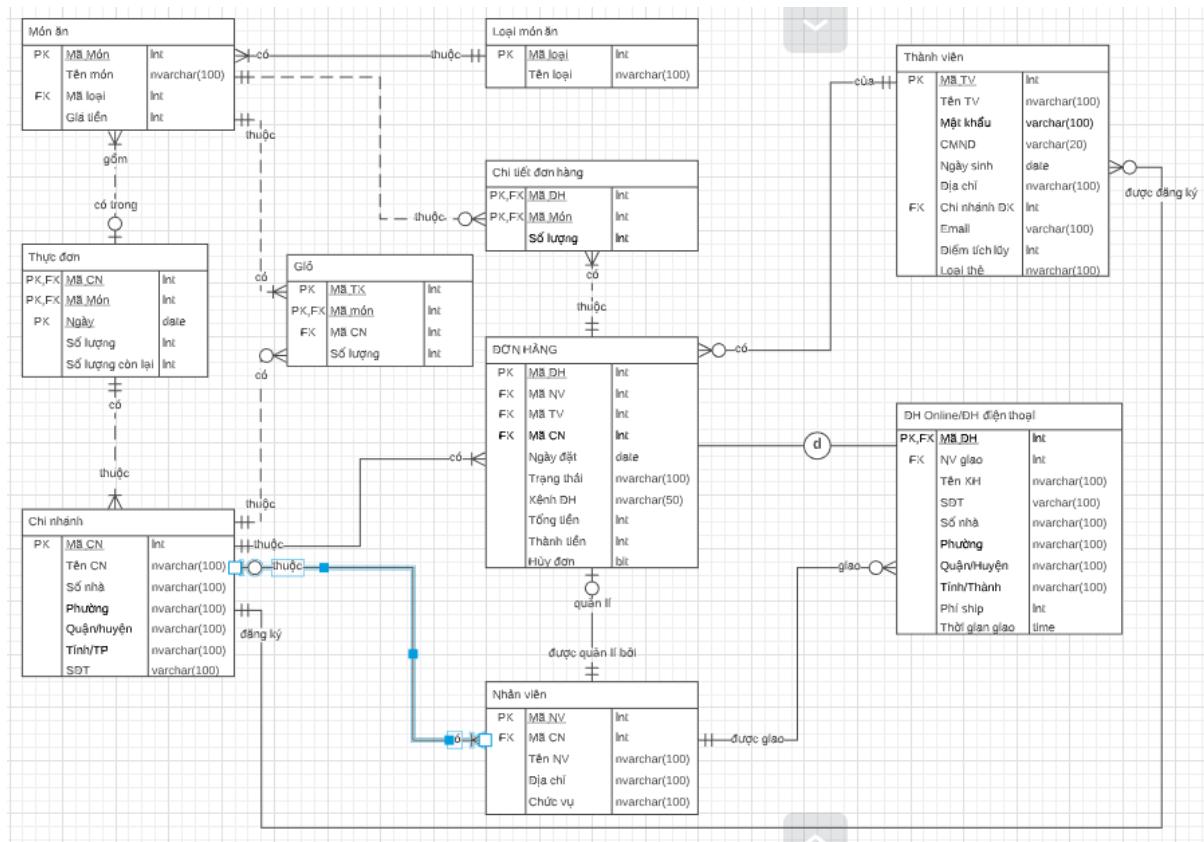
Với đồ án Hương Việt restaurant và kết quả thiết kế CSDL ở BTTH1, hãy thực hiện các công việc sau:

- Xác định các chức năng nhóm sẽ cài đặt và có khả năng xảy ra tranh chấp đồng thời/deadlock.
- Viết các stored procedure, function cho các chức năng trên.
- Xác định các tranh chấp đồng thời và phương án xử lý tranh chấp/deadlock.
- Thiết kế giao diện và xây dựng ứng dụng để minh họa:
 - o Các tình huống tranh chấp/deadlock đã xác định
 - o Kết quả sau khi sử dụng các phương án xử lý các tranh chấp/deadlock

B. KẾT QUẢ

I. Thiết kế cơ sở dữ liệu

1. Lược đồ cơ sở dữ liệu



Hình 1.1

2. Mô tả cơ sở dữ liệu

Tên bảng	Tên thuộc tính	Kiểu dữ liệu	Mô tả
CHINHANH	<u>MACN</u>	Int	Mã chi nhánh, mỗi chi nhánh có duy nhất một mã chi nhánh.
	TENCN	Nvarchar(100)	Tên của chi nhánh.
	SONHA	Nvarchar(100)	Số nhà (Địa chỉ của chi nhánh).
	PHUONG	Nvarchar(100)	Phường (Địa chỉ của chi nhánh).
	QUAN_HUYEN	Nvarchar(100)	Quận/ Huyện (Địa chỉ của chi nhánh).
	TINH_TP	Nvarchar(100)	Tỉnh / Thành phố (Địa chỉ của chi nhánh).
	SDT	Varchar(100)	Số điện thoại để liên lạc của chi nhánh.
MONAN	<u>MAMON</u>	Int	Mã món ăn, mỗi món ăn thuộc nhà hàng có một mã món duy nhất.
	TENMON	Nvarchar(100)	Tên của món ăn.
	<u>MALOAI</u>	Int	Mã loại món ăn, mỗi món ăn chỉ có 1 mã loại.
	GIATIEN	Int	Giá tiền của một món ăn.
LOAIMON	<u>MALOAI</u>	Int	Mã loại món ăn, mỗi loại có duy nhất một mã loại, mỗi loại có 1 hoặc nhiều món ăn.

	TENLOAI	Nvarchar(100)	Tên của loại món ăn.
THUCDON	<u>MACN</u>	Int	Thực đơn thuộc về chi nhánh.
	<u>MAMON</u>	Int	Các món ăn có trong thực đơn.
	<u>NGAY</u>	Date	Ngày thực đơn được sử dụng.
	SOLUONG	Int	Số lượng ban đầu của mỗi món.
	SL_CONLAI	Int	Số lượng còn lại của mỗi món.
DONHANG	<u>MADH</u>	Int	Mã đơn hàng, mỗi đơn hàng có duy nhất một mã đơn hàng.
	<u>MANV</u>	Int	Nhân viên lập đơn hàng.
	<u>MATV</u>	Int	Thành viên đặt đơn hàng.
	<u>MACN</u>	Int	Đơn hàng của chi nhánh nào.
	NGAYDAT	Date	Ngày lập đơn hàng
	TRANGTHAI	Nvarchar(100)	Trạng thái của đơn hàng.
	KENHDH	Nvarchar(100)	Kênh đặt hàng (Trực tiếp, Điện thoại, Online)

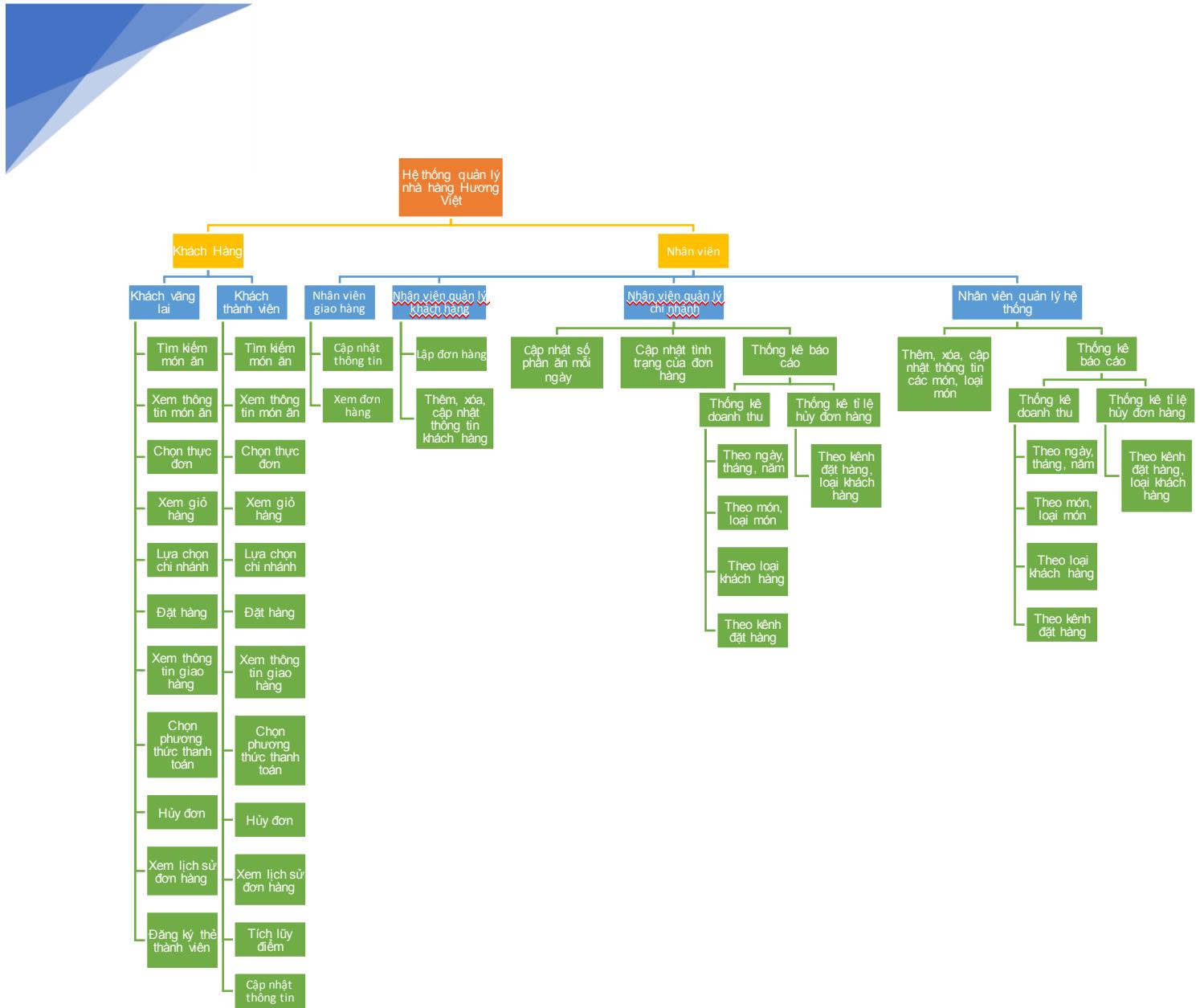
	THANHTIEN	Int	Tổng tiền của tất cả món ăn có trong đơn hàng dựa trên số lượng mỗi món.
	TONGTIEN	Float	Tổng tiền của đơn hàng (bao gồm: thành tiền món ăn, phí ship, giảm giá,...)
	PTTT	Nvarchar(100)	Phương thức thanh toán.
	HUYDON	Bit	Hủy đơn hàng.
DHTRUCTUYEN	<u>MADH</u>	Int	Mã đơn hàng trực tuyến (thuộc kênh đặt hàng điện thoại hoặc online).
	<u>NVGIAOHANG</u>	Int	Mã nhân viên giao hàng.
	TENKH	Nvarchar(100)	Tên khách hàng nhận hàng.
	SDT	Varchar(100)	Số điện thoại của khách hàng nhận hàng.
	SONHA	Nvarchar(100)	Số nhà của khách hàng nhận hàng.
	PHUONG	Nvarchar(100)	Phường của khách hàng nhận hàng.
	QUAN_HUYEN	Nvarchar(100)	Quận/ huyện của khách hàng nhận hàng.

	TINH_TP	Nvarchar(100)	Tỉnh/ thành phố của khách hàng nhận hàng.
	CHIPHI	Int	Chi phí giao hàng.
	TGGIAO	Time	Thời gian giao hàng dự kiến.
CHITIETDONHANG	<u>MADH</u>	Int	Mã đơn hàng.
	<u>MAMON</u>	Int	Mã món ăn.
	SL	Int	Số lượng mỗi món ăn.
THANHVIEN	<u>MATV</u>	Int	Mã thành viên, mỗi thành viên khi đăng ký thẻ/ tài khoản sẽ được một mã thành viên duy nhất.
	TENTV	Nvarchar(100)	Tên thành viên đăng ký thẻ/ tài khoản.
	MATKHAU	Varchar(100)	Mật khẩu của thẻ/ tài khoản thành viên.
	CMND	Varchar(20)	Chứng minh nhân dân, đồng thời cũng là tên đăng nhập cho tài khoản của thành viên.
	NGAYSINH	Date	Ngày sinh của thành viên.
	DIACHI	Nvarchar(100)	Địa chỉ của thành viên.
	<u>CHINHANHDK</u>	Int	Chi nhánh thành viên đăng ký thẻ/ tài khoản.

	EMAIL	Nvarchar(100)	Email của thành viên.
	DIEMTICHLUY	Int	Điểm tích lũy của thành viên (sau các lần đặt đơn hàng).
	LOAITHE	Nvarchar(100)	Loại thẻ của thành viên (để nhận ưu đãi, giảm giá).
NHANVIEN	<u>MANV</u>	Int	Mã nhân viên, mỗi nhân viên có một mã nhân viên duy nhất.
	<u>MACN</u>	Int	Chi nhánh nhân viên làm việc.
	TENNVIEN	Nvarchar(100)	Tên của nhân viên.
	DIACHI	Nvarchar(100)	Địa chỉ của nhân viên.
	CHUCVU	Nvarchar(100)	Chức vụ của nhân viên.
GIO	<u>MATK</u>	Int	Tài khoản của thành viên.
	<u>MAMON</u>	Int	Món ăn được chọn.
	<u>MACN</u>	Int	Chi nhánh đặt đơn hàng.
	SL	Int	Số lượng món ăn đã chọn.

II. Đặc tả chức năng

- ❖ Đặc tả chức năng theo dạng cây:



III. Mô tả các stored procedure

Tên thủ tục	Mục đích	Input	Output
sp_THEMTHANHVIENTHONG	Khách hàng đăng ký thẻ thành viên.	@TENTV, @MATKHAU, @CMND, @NGSINH, @CHINHANHDK, @DIACHI, @EMAIL	Dữ liệu mới được thêm vào bảng THANHVIEN.
SP_DangNhap	Thành viên đăng nhập vào hệ thống.	@user, @pass	Thông tin của thành viên.

sp_Khach	Nhân viên thêm thông tin của khách hàng để làm thẻ thành viên.	@tentv, @cmnd, @ngay_sinh, @chinhanhdk, @DIACHI, @email, @pass	Dữ liệu mới được cập nhật vào bảng THANHVIEN.
sp_SuaKhach	Thành viên cập nhật thông tin của mình.	@matv, @tentv, @cmnd, @ngay_sinh, @dia_chi,@chinhanhdk, , @email	Dữ liệu mới được cập nhật vào bảng THANHVIEN
sp_XoaKhach	Xóa thành viên.	@matv	Dữ liệu của thành viên sẽ bị xóa khỏi bảng THANHVIEN
sp_UPTHETV3	Cập nhật mật khẩu mới cho thành viên.	@MATV, @MATKHAU, @MATKHAUMOI	Mật khẩu được cập nhật lại.
sp_CapNhatSoPhanAn	Để cập nhật số lượng của một món tại một chi nhánh trong một ngày nhất định.	@MACN, @MAMON, @NGAY, @SOLUONG, @SL_CON	Cập nhật dữ liệu vào bảng thực đơn.
sp_ThongKeTheoNgayCN	Thống kê doanh thu theo từng ngày của một chi nhánh nhất định.	@MACN	Gồm ngày và doanh thu trong ngày đó.
sp_ThongKeTheoThangCN	Thống kê doanh thu theo từng tháng của một chi nhánh nhất định.	@MACN	Gồm tháng, năm và doanh thu trong tháng đó.
sp_ThongKeTheoNamCN	Thống kê doanh thu theo từng năm của	@MACN	Gồm năm và doanh thu trong năm đó.

	một chi nhánh nhất định.		
sp_ThongKeTheoMonCN	Thống kê doanh thu theo món của một chi nhánh nhất định.	@MACN	Gồm món và doanh thu của món đó.
sp_ThongKeTheoLoaiMonCN	Thống kê doanh thu theo loại món của một chi nhánh nhất định.	@MACN	Gồm loại món và doanh thu của loại món đó.
sp_TheoKenhDatHangCN	Thống kê doanh thu theo kênh đặt hàng của một chi nhánh nhất định.	@MACN	Gồm kênh đặt hàng và doanh thu của kênh đó.
sp_ThongKeTheoLoaiKhachHangCN	Thống kê doanh thu theo loại khách hàng của một chi nhánh nhất định.	@MACN	Gồm 2 cột là khách vãng lai và thành viên chứa doanh thu của từng loại.
sp_ThongKeTheoThangTong	Thống kê doanh thu theo từng tháng của toàn chi nhánh.		Gồm tháng, năm và doanh thu trong tháng đó.
sp_ThongKeTheoNamTong	Thống kê doanh thu theo từng năm của toàn chi nhánh.		Gồm năm và doanh thu trong năm đó.
sp_ThongKeTheoMonTong	Thống kê doanh thu theo từng món của toàn chi nhánh.		Gồm tên món và doanh thu của món đó.
sp_ThongKeTheoLoaiMonTong	Thống kê doanh thu theo từng loại món của toàn chi nhánh.		Gồm tên loại món và doanh thu của loại món đó.

sp_ThongKeTheoNgayTon g	Thống kê doanh thu theo từng ngày của toàn chi nhánh.		Gồm ngày và doanh thu trong ngày đó.
sp_TheoKenhDatHangTon g	Thống kê doanh thu theo kênh đặt hàng của toàn chi nhánh.		Gồm kênh đặt hàng và doanh thu của kênh đó.
sp_ThongKeTheoLoaiKha chHangTong	Thống kê doanh thu theo loại khách hàng của toàn chi nhánh.		Gồm khách vãng lai và thành viên.
sp_XemThucDon	Để xem thực đơn của một chi nhánh trong một ngày.	@MACN, @Ngay	Gồm mã món, tên món, tên loại và số lượng.
sp_TiLeHuyDonTheoKenh	Tính tỉ lệ hủy đơn theo các kênh đặt hàng trong một chi nhánh.	@MACN	Gồm 3 cột là tỉ lệ hủy Online, tỉ lệ hủy điện thoại và tỉ lệ hủy trực tiếp.
sp_TiLeHuyDonTheoLoai KH	Tính tỉ lệ hủy đơn theo loại khách hàng trong một chi nhánh.	@MACN	Gồm 2 cột là tỉ lệ hủy theo khách vãng lai và tỉ lệ hủy theo thành viên.
sp_TiLeHuyDonTheoKenh Tong	Tỉ lệ hủy đơn theo kênh đặt hàng của toàn chi nhánh.		Gồm Online, điện thoại và trực tiếp.
sp_TiLeHuyDonTheoLoai KHTong	Tỉ lệ hủy đơn theo loại khách của toàn chi nhánh.		Gồm khách vãng lai và thành viên.
sp_ThemMon	Thêm món vào bảng MONAN.	@TENMON, @MALOAI, @GIA	Thêm dữ liệu vào bảng MONAN.
sp_XoaMon	Xóa món khỏi bảng MONAN.	@MAMON	Xóa dữ liệu khỏi bảng MONAN.

sp_SuaMon	Sửa món trong bảng MONAN.	@MAMON, @TENMON, @MALOAI, @GIA	Sửa dữ liệu một món trong bảng MONAN.
sp_ThemLoaiMon	Thêm một loại món vào bảng LOAIMON.	@TENLOAI	Thêm dữ liệu vào bảng LOAIMON.
sp_XoaLoaiMon	Xóa một loại món khỏi bảng LOAIMON.	@MALOAI	Xóa dữ liệu khỏi bảng LOAIMON.
sp_InDH	Nhân viên thêm đơn hàng mới.	@MANV, @MATV, @MACN, @KENHDH, @PTTT	Một đơn hàng mới được thêm vào bảng DONHANG.
sp_UpDH	Nhân viên cập nhật tình trạng của đơn hàng.	@MADH, @TRANGTHAI	Toàn bộ thông tin của đơn hàng vừa được cập nhật trạng thái.
sp_UpDH_HuyDon	Khách hàng hoặc nhân viên hủy đơn.	@MADH, @HUYDON	Toàn bộ thông tin của đơn hàng vừa bị hủy.
sp_HienGio	Khách hàng xem giờ hàng của mình.	@MATK	Tên, số lượng, giá tiền các món ăn trong giờ hàng.
Sp_InCTDH	Nhân viên hoặc khách hàng thêm món ăn vào đơn hàng.	@MADH, @MAMON, @SL	Thông tin các món ăn có trong đơn hàng (bao gồm cả món vừa được thêm).
Sp_DelCTDH	Nhân viên hoặc khách hàng xóa món ăn trong đơn hàng.	@MADH, @MAMON	Thông tin các món ăn có trong đơn hàng.

Sp_InDHTT	Nhân viên thêm đơn hàng trực tuyến.	@MADH, @NVGIAOHANG, @TENKH, @SDT, @SONHA, @PHUONG, @QUAN_HUYEN, @TINH_TP	Thông tin của đơn hàng vừa được thêm.
Sp_UpDHTT	Nhân viên cập nhật nhân viên giao hàng của đơn hàng trực tuyến.	@MADH, @MANV	Đơn hàng được thay đổi nhân viên giao hàng.
sp_ThemGio	Thêm một món ăn mới vào giỏ hàng hoặc tăng số lượng món ăn đã có sẵn trong giỏ hàng.	@matk, @ma_mon, @ma_cn, @s1	Một món ăn mới được thêm vào giỏ hàng hoặc được tăng số lượng.
sp_XoaGio	Xóa một món ăn trong giỏ hàng hoặc giảm số lượng món ăn đã có sẵn trong giỏ hàng.	@matk, @ma_mon, @s1	Một món ăn trong giỏ hàng bị xóa hoặc bị giảm số lượng.
sp_DonHangCaNhan	Hiển thị các đơn hàng của thành viên.	@MATV	Tất cả đơn hàng cá nhân của thành viên.
sp_ThemDonHangOnline	Thêm đơn mới với dữ liệu từ giỏ hàng.		Đơn hàng mới được thêm với tất cả các món ăn được chọn từ giỏ hàng tương ứng.
sp_TinhTrangDon	Hiển thị danh sách trạng thái các đơn hàng.		Tất cả đơn hàng với mã đơn hàng cùng với trạng thái tương ứng của đơn.
sp_ThemMonThucDon	Thêm một món mới vào thực đơn.	@MACN, @MAMON, @NGAY, @SOLUONG, @SL_CON	Một món mới được thêm vào thực đơn.

sp_XoaMonThucDon	Xóa một món ăn trong thực đơn.	@MACN, @MAMON, @NGAY	Một món ăn trong thực đơn bị xóa.
------------------	--------------------------------	----------------------------	-----------------------------------

IV. Mô tả kịch bản lỗi tranh chấp đồng thời/ deadlock

1. Mô tả thủ tục tranh chấp

Tên lỗi	Kịch bản lỗi	Xử lý lỗi	Sinh viên thực hiện
Dirty Read (Các ví dụ đều được thiết lập mức cô lập Read Uncommitted cho giao tác T2)	a. Giả sử món ăn số 1 tại chi nhánh 1 đã được đặt hết và có một khách hàng A huỷ món ăn số 1 tại chi nhánh 1 nhưng chưa commit giao tác, tại thời điểm đó có 1 khách hàng B tới tìm món số 1 tại chi nhánh 1 và thấy số lượng còn lại là 4, sau đó khách hàng A đổi ý không huỷ món nữa. Kết quả là khách hàng B đặt được món trong khi nhà hàng không còn món đó nữa.		1712415
	b. Giả sử, do nguyên liệu để chế biến món có mã món 1 ở chi nhánh 1 không đủ và chưa kịp cung ứng. Nên quản lý chi nhánh 1 đã cập nhật số lượng còn lại của món 1 chỉ còn 10 phần nhưng chưa commit. Trong lúc này thì có một khách hàng online xem thực đơn ở chi nhánh 1 thì thấy với mã món số 1 thì số lượng còn lại là 10 nhưng cần đặt 11 phần nên không đặt nữa. Sau đó đã gặp sự cố nên giao tác của quản lý đã rollback không cập nhật thành công.	Thiết lập mức cô lập Read Committed.	1712662
	c. Giả sử nhân viên 1 thêm món vào thực đơn, nhân viên 2 xem thực đơn thì sẽ bị đọc lỗi nếu giao tác T1 không commit mà rollback lại.		1712671

	d. Quản lý hệ thống muốn cập nhật giá tiền món ăn 1 thành 25 000 nhưng cập nhật nhầm món ăn 2, khi đó có khách hàng online vào xem món ăn 2 thì thấy có giá mới là 25 000. Sau đó, quản lý phát hiện cập nhật nhầm và rollback, giá tiền của món 2 trở lại thành giá ban đầu.		1712769
	e. Giả sử khách hàng số 1 vào xem và thấy số lượng còn lại của món số 1 tại chi nhánh 1 là 1, người này tiến hành đặt hàng nhưng chưa commit thao tác. Trong lúc này thì khách hàng số 2 cùng vào xem món 1 tại chi nhánh 1 và thấy số lượng còn lại là 0, người này không đặt nữa. Khách 1 không đặt món nữa và rollback lại, lúc này khách hàng số 2 đã đọc dữ liệu sai.		1712899
Unrepeatable Read	a. Khách hàng đang xem giá số lượng món ăn số 1 ở chi nhánh 1, có khách hàng khác đặt món. Quản lý tiến hành cập nhật số lượng thì khách hàng đang xem sẽ thấy số lượng thay đổi.		1712415
	b. Khách hàng 1 thực hiện xem thực đơn của chi nhánh 1, thì thấy số lượng còn lại của món 1 là 50. Sau đó có khách hàng 2 tới đặt 5 phần món 1 ở chi nhánh 1 và được nhân viên lẽ tân tiếp nhận đơn. Khách hàng 1 thực hiện xem thực đơn lại một lần nữa thì thấy số lượng còn lại của món 1 là 45.	Thiết lập mức độ lặp Reentrant Read.	1712662

	c. T1 xem trong thực đơn, sau đó T2 xóa trong phần T1 đã xem sau đó T1 xem lại thì kết quả không giống nhau		1712671
	d. Khách hàng 1 đang muốn đặt món ăn 1 ở chi nhánh 1 với số lượng là 40, khi họ vào xem thực đơn thì thấy món ăn ban đầu là 50 và dự định tiến hành đặt hàng, nhưng có có khách hàng 2 đặt hàng nhanh hơn, và số lượng còn lại của món ăn 1 thay đổi thành 30. Khi khách hàng 1 xem lại thì thấy số lượng đã bị thay đổi và không đủ để cung cấp.		1712769
	e. Khách hàng có mã thành viên = 1 đang xem thông tin tài khoản của mình ở chi nhánh 1, thì có người khác cũng vào tài khoản này và sửa thông tin DIACHI. Khách hàng thực hiện xem thông tin lại một lần nữa thì thông tin ban đầu đã bị thay đổi.		1712899
Phantom	a. Quản lý chi nhánh thống kê những thành viên của mình, tại thời điểm đó thì nhân viên lễ tân lại chèn thêm 1 thành viên mới.		1712415
	b. Giả sử khách hàng xem thực đơn của chi nhánh 1 thấy chỉ có món 1. Sau đó, quản lý công ty có thêm món 2 vào thực đơn. Khi khách hàng xem lại thực đơn thì thấy xuất hiện thêm món 2.	Thiết lập mức cô lập Serializable	1712662
	c. Giả sử T1 thống kê vào ngày hôm nay tại chi nhánh 1. Sau đó T2 insert vào bảng đơn hàng. T1 thống kê lại		1712671

	<p>thì thấy thêm dữ liệu T2 vừa thêm vào dù chưa commit. Lỗi phantom.</p>		
	<p>d. Nhân viên quản lý chi nhánh 1 đang xem các thành viên loại Gold thuộc chi nhánh 1 thấy có 20 thành viên, sau đó có đơn hàng của thành viên 23 vừa hoàn thành xong và thành viên đó được hệ thống nâng cấp lên loại Gold. Nhân viên quản lý xem lại thì thấy có thêm một thành viên được loại Gold.</p>		1712769
	<p>e. Giả sử nhân viên quản lý khách hàng đang xem lại thông tin của tất cả khách hàng đã đăng ký thành viên trong chi nhánh 2.</p> <p>Trong lúc này, một khách hàng khác vào và đăng ký thành viên trong chi nhánh 2 bằng kênh online. Khi nhân viên xem lại thì thấy số lượng khách hàng đăng ký thành viên trong chi nhánh 2 bị thay đổi.</p>		1712899
Lost Update	<p>a. Giả sử khách hàng A (ban đầu đã mua món số 2) đặt thêm hàng món số 2 tại chi nhánh 1 với số lượng là 1, Lúc này có 2 nhân viên quản lý khách hàng truy cập vào tài khoản để cập nhật lại chi tiết đơn hàng. Nhân viên 1 tăng số lượng món lên 1, nhân viên 2 cũng tăng số lượng lên 1. Nhân viên 1 nếu lần sau vào xem sẽ thấy không giống thông tin mình đã sửa.</p>	<p>Thiết lập mức độ Repeatable Read hoặc Serializable nhưng sẽ tạo thành Deadlock</p>	1712415
	<p>b. Giả sử quản lý chi nhánh 1 vào xem thực đơn chi nhánh 1 sau đó cập nhật số phần còn lại của món 1 là 45. Cùng lúc đó quản lý công ty vào xem</p>		1712662

	<p>thực đơn chi nhánh 1, sau đó cập nhật số phần còn lại của món 1 là 40. Nếu sau đó quản lý chi nhánh vào xem lại thực đơn thì sẽ không hiện thông tin như mình vừa cập nhật.</p>		
	c. T1 đăng nhập và thay đổi giá trị thẻ, sau đó T2 đăng nhập và cũng thay đổi giá trị thẻ. Qua đó update của T1 bị mất		1712671
	d. Một nhân viên quản lý khán hàng 1 của chi nhánh 1 tạo đơn hàng và update số lượng món 1 trong bảng thực đơn từ 30 thành 19, cùng lúc đó một nhân viên quản lý khán hàng 2 của chi nhánh 1 tạo đơn hàng khác và update số lượng món 1 trong bảng thực đơn từ 30 thành 17. Khi nhân viên quản lý khán hàng 2 xem lại thực đơn thì thấy số lượng món ăn 1 sai.		1712769
	e. Giả sử có 2 người đồng thời truy cập vào tài khoản thành viên 1. Người số 1 vào cập nhật lại địa chỉ là 'số 7, Nguyễn Hữu Cánh, phường 3, quận 2, TPHCM', người số 2 vào cũng cập nhật lại địa chỉ '370, Nguyễn Văn Cừ, phường 10, quận 5, TPHCM', người thứ 1 nếu lần sau vào xem lại thông tin thì sẽ không giống với thông tin ban đầu mình đã sửa nữa.		1712899
Deadlock	a. Giả sử khách hàng A (ban đầu đã mua món số 2) đặt thêm hàng món số 2 tại chi nhánh 1 với số lượng là 1, Lúc này có 2 nhân viên quản lý khách hàng truy cập vào tài khoản để cập nhật lại chi tiết đơn hàng. Nhân viên	Dùng các thuật toán Wait-Die, Wound-Wait, hoặc hủy một	1712415

	<p>1 tăng số lượng món lên 1, nhân viên 2 cũng tăng số lượng lên 1. Nhân viên 1 nếu lần sau vào xem sẽ thấy không giống thông tin mình đã sửa. để xảy ra deadlock ta xét mức cô lập SERIALAZABLE</p>	trong hai giao tác.	
	<p>b. Giả sử có 2 người quản lý công ty. Người thứ nhất cập nhật số phần ăn của món 1 ở chi nhánh 1 sau đó tiến hành xem thực đơn chi nhánh 2 cần chỉnh sửa gì không. Cùng lúc đó người thứ 2 cập nhật số phần ăn của món 1 ở chi nhánh 2 và sau đó tiến hành xem thực đơn chi nhánh 1. Lúc này giao tác của người thứ 2 bị rollback và thực hiện lại theo cơ chế xử lí Deadlock của SQL.</p>	1712662	
	<p>c. Giả sử có hai người quản lý công ty, T1 thêm món vào thực đơn chi nhánh 1 và xem thực đơn chi nhánh 2, T2 thêm món vào thực đơn chi nhánh 2 và xem món chi nhánh 1.</p>	1712671	
	<p>d. Nhân viên quản lý khách hàng xem thông tin đơn hàng 3 và sắp cập nhật trạng thái cho đơn hàng 3 từ 'Tiếp nhận' sang 'Đang chuẩn bị', trong lúc đó thành viên 1 muốn hủy đơn hàng 3 đăng nhập vào xem thấy trạng thái vẫn là 'Tiếp nhận' và thực hiện hủy đơn => Deadlock xảy ra.</p> <p>Ở đây, ta set cơ chế khóa Serializable cho cả hai giao tác.</p>	1712769	
	<p>e. Giả sử có 2 người đồng thời truy cập vào tài khoản thành viên 1. Người số 1 vào cập nhật lại địa chỉ là 'số 7, Nguyễn Hữu Cánh, phường 3,</p>	1712899	

	quận 2, TPHCM', người số 2 vào cũng cập nhật lại địa chỉ '122, Nguyễn Văn Cừ, phường 10, quận 5, TPHCM', người thứ 1 nếu lần sau vào xem lại thông tin thì sẽ không giống với thông tin ban đầu mình đã sửa nữa.		
--	--	--	--

2. Chi tiết các thủ tục tranh chấp

2.1. Dirty Read

a. Lỗi a

❖ **Sinh viên thực hiện:** Nguyễn Thị Ngọc Hân – 1712415

i. Kịch bản lỗi

Giả sử món ăn số 1 tại chi nhánh 1 đã được đặt hết và có một khách hàng A huỷ món ăn số 1 tại chi nhánh 1 nhưng chưa commit giao tác, tại thời điểm đó có 1 khách hàng B tới tìm món số 1 tại chi nhánh 1 và thấy số lượng còn lại là 4, sau đó khách hàng A đổi ý không huỷ món nữa. Kết quả là khách hàng B đặt được món trong khi nhà hàng không còn món đó nữa. . Ở ví dụ này, để xảy ra lỗi thì giả lập mức độ lỗi cho giao tác T2 là Read Uncommitted.

T1	T2
<pre>BEGIN TRAN UPDATE THUCDON SET SL_CONLAI= 4 WHERE MACN = 1 AND MAMON=1 AND NGAY='2019-30-11' waitfor delay'00:00:05'</pre>	<pre>BEGIN TRAN SET TRAN ISOLATION LEVEL READ UNCOMMITTED SELECT SOLUONGCONLAI FROM THUCDON WHERE MACN = 1 AND</pre>

ROLLBACK	MAMON=1 AND NGAY='2019-30-11'
	COMMIT

ii. Chạy lỗi trên SQL

Đoạn test case của trường hợp Dirty Read:

```

1
2
3 --THỰC HIỆN: NGUYỄN THỊ NGỌC HÂN
4   --MSSV: 1712415
5 BEGIN TRAN
6 UPDATE THUCDON
7 SET SL_CONLAI= 4 WHERE MACN = 1 AND MAMON=1 AND NGAY='2019-11-30'
8 waitfor delay'00:00:15'
9
10
11
12 ROLLBACK
13
14
15

```

```

1
2 --THỰC HIỆN: NGUYỄN THỊ NGỌC HÂN
3   --MSSV: 1712415
4 BEGIN TRAN
5 SET TRAN ISOLATION LEVEL READ UNCOMMITTED
6 SELECT SL_CONLAI
7 FROM THUCDON
8 WHERE MACN = 1 AND MAMON=1 AND NGAY='2019-11-30'
9
10
11 COMMIT
12

```

Hình 2.1.a.1

Chạy giao tác 1 đến lệnh waitfor delay '00:00:15', sau đó chạy hết giao tác 2 và ta có kết quả:

```

SQLQuery1.sql - D...09AIR6\Admin (51)* x SQLQuery2.sql - D...09AIR6\Admin (54)*
1
2
3 --THỰC HIỆN: NGUYỄN THỊ NGỌC HÂN
4   --MSSV: 1712415
5 BEGIN TRAN
6 UPDATE THUCDON
7 SET SL_CONLAI= 4 WHERE MACN = 1 AND MAMON=1 AND NGAY='2019-11-30'
8 waitfor delay'00:00:15'
9
10
11
12 ROLLBACK
13
14
15

```

```

1
2 --THỰC HIỆN: NGUYỄN THỊ NGỌC HÂN
3   --MSSV: 1712415
4 BEGIN TRAN
5 SET TRAN ISOLATION LEVEL READ UNCOMMITTED
6 SELECT SL_CONLAI
7 FROM THUCDON
8 WHERE MACN = 1 AND MAMON=1 AND NGAY='2019-11-30'
9
10
11 COMMIT
12

```

Messages

(1 row(s) affected)

Results

SL_CONLAI
4

Hình 2.1.a.2

⇒ Giao tác 2 đã đọc dữ liệu mà giao tác 1 đã cập nhật nhưng chưa rollback với nội dung Số lượng còn lại của món ăn có mã 1 là 4.

Sau khi Giao tác 1 rollback, thử truy vấn lại thì số lượng còn lại đã trở lại 76.

```

1
2
3 --THỰC HIỆN: NGUYỄN THỊ NGỌC HÂN
4 |--MSSV: 1712415
5 BEGIN TRAN
6 UPDATE THUCDON
7 SET SL_CONLAI= 4 WHERE MACN = 1 AND MAMON=1 AND NGAY='2019-11-30'
8 waitfor delay'00:00:15'
9
10
11
12
13 ROLLBACK
14
15 SELECT SL_CONLAI FROM THUCDON WHERE MACN = 1 AND MAMON=1 AND NGAY='2019-11-30'

```

Results

SL_CONLAI
76

Hình 2.1.a.3

iii. Giải pháp: Set cơ chế khóa Read Committed

T1	KHOÁ	T2	KHOÁ
BEGIN TRAN UPDATE THUCDON SET SL_CONLAI= 4 WHERE MACN = 1 AND MAMON=1 AND NGAY='2019-30-11' waitfor delay'00:00:05'	T1 xin khóa X(THUCDON) SQL: cấp X(THUCDON) và giữ X cho đến hết giao tác	Set READ COMMITTED	
		BEGIN TRAN SELECT SOLUONGCONLAI FROM THUCDON WHERE MACN = 1 AND MAMON=1 AND NGAY='2019-30-11'	T2 xin khóa S(THUCDON) SQL: rỗng
ROLLBACK		COMMIT	

Chạy giao tác 1 đến lệnh waitfor delay '00:00:15', sau đó chạy hết giao tác 2, lúc này T2 không đọc được dữ liệu mà T1 thay đổi do T1 vẫn chưa kết thúc giao tác.

```

1
2
3 --THỰC HIỆN: NGUYỄN THỊ NGỌC HÂN
4 |--MSSV: 1712415
5 BEGIN TRAN
6 UPDATE THUCDON
7 SET SL_CONLAI= 4 WHERE MACN = 1 AND MAMON=1 AND NGAY='2019-11-30'
8 waitfor delay'00:00:15'
9
10
11
12 ROLLBACK
13
14
15 SELECT SL_CONLAI FROM THUCDON WHERE MACN = 1 AND MAMON=1 AND NGAY='2019-11-30'

SQLQuery1.sql - D...09AIR6\Admin (51)* < SQLQuery3.sql - DES...n (54) Executing...
1
2 |--THỰC HIỆN: NGUYỄN THỊ NGỌC HÂN
3 |--MSSV: 1712415
4 BEGIN TRAN
5 SET tran isolation level read committed
6 SELECT SL_CONLAI
7 FROM THUCDON
8 WHERE MACN = 1 AND MAMON=1 AND NGAY='2019-11-30'
9
10
11
12 COMMIT
13
14
15

Results
```

(1 row(s) affected)

Hình 2.1.a.4

Sau khi T1 rollback, T2 lập tức đọc được dữ liệu ban đầu.

```

SQLQuery1.sql - D...09AIR6\Admin (51)* × SQLQuery3.sql - D...09AIR6\Admin (54)*
1
2 --THỰC HIỆN: NGUYỄN THỊ NGỌC HÂN
3 --MSSV: 1712415
4 BEGIN TRAN
5 UPDATE THUDON
6 SET SL_CONLAI= 4 WHERE MACN = 1 AND MAMON=1 AND NGAY='2019-11-30'
7 waitfor delay '00:00:15'
8
9
10
11
12 ROLLBACK
13
14
15 SELECT SL_CONLAI FROM THUDON WHERE MACN = 1 AND MAMON=1 AND NGAY='2019-11-30'

100 % Messages
Command(s) completed successfully.

SQLQuery1.sql - D...09AIR6\Admin (51)* × SQLQuery3.sql - D...09AIR6\Admin (54)*
1
2 --THỰC HIỆN: NGUYỄN THỊ NGỌC HÂN
3 --MSSV: 1712415
4 BEGIN TRAN
5 SET tran isolation level read committed
6 SELECT SL_CONLAI
7 FROM THUDON
8 WHERE MACN = 1 AND MAMON=1 AND NGAY='2019-11-30'
9
10
11 COMMIT
12

100 % Results
SL_CONLAI
1 76

```

Hình 2.1.a.5

b. Lỗi b

❖ **Sinh viên thực hiện:** Lai Gia Phú – 1712662

i. Kích bản lỗi

Giả sử, do nguyên liệu để chế biến món có mã món 1 ở chi nhánh 1 không đủ và chưa kịp cung ứng. Nên quản lý chi nhánh 1 đã cập nhật số lượng còn lại của món 1 chỉ còn 10 phần nhưng chưa commit. Trong lúc này thì có một khách hàng online xem thực đơn ở chi nhánh 1 thì thấy với mã món số 1 thì số lượng còn lại là 10 nhưng cần đặt 11 phần nên không đặt nữa. Sau đó đã gặp sự cố nên giao tác của quản lý đã rollback không cập nhật thành công. Ở ví dụ này, để xảy ra lỗi thì giả lập mức cô lập cho giao tác T2 là Read Uncommitted.

T1	T2
<pre> BEGIN TRAN EXEC sp_CapNhatSoPhanAn 1,1,'2019-01-01',50,10 Waitfor delay '00:00:10' ROLLBACK TRAN </pre>	<pre> BEGIN TRAN set tran isolation level READ UNCOMMITTED EXEC sp_XemThucDon 1,'2019-01- 01' COMMIT TRAN </pre>

ii. Chạy lỗi trên SQL

Đoạn test case của trường hợp Dirty Read:

Chạy giao tác 1 đến lệnh waitfor delay '00:00:10', sau đó chạy hết tác 2 và ta có kết quả:

```
-- T1
BEGIN TRAN
EXEC sp_CapNhatSoPhanAn 1,1,'2019-01-01',50,10
Waitfor delay '00:00:10'
ROLLBACK TRAN

-- T2
BEGIN TRAN
set tran isolation level READ UNCOMMITTED
EXEC sp_XemThucDon 1,'2019-01-01'
COMMIT TRAN
```

Messages

(1 row affected)
Sửa thành công
Completion time: 2020-01-01T17:27:32.5755200+07:00

Results

MAMON	TENMON	TENLOAI	SL_CONLAI	SOLUONG
1	PHỞ	MÓN NƯỚNG	10	50
2	BÚN CHĀ	MÓN NƯỚNG	50	50
3	BÚN NEM NƯỚNG	MÓN NƯỚNG	50	50
4	CƠM CHIÊN	CƠM	50	50
5	BÁNH XÈO	BÁNH MẶN	50	50
6	BÁNH BÈO	BÁNH MẶN	50	50
7	BÁNH KHỌT	BÁNH MẶN	50	50
8	CHE BUỒI	CHE	50	50
9	CHE HẠT SEN	CHE	50	50
10	CHE KHÚC BẠCH	CHE	50	50

Hình 2.1.b.1

⇒ Giao tác 2 đã đọc dữ liệu mà giao tác 1 đã cập nhật nhưng chưa rollback với nội dung Số lượng còn lại của món ăn có mã 1 là 10.

Sau khi Giao tác 1 rollback, thử truy vấn lại thì số lượng còn lại đã trở lại 50.

The screenshot shows a SQL Server Management Studio window. The query window contains the following T-SQL code:

```
--T2
BEGIN TRAN
set tran isolation level READ UNCOMMITTED
EXEC sp_XemThucDon 1, '2019-01-01'
COMMIT TRAN
```

The results grid displays a list of 10 menu items (MAMON) with their details:

	MAMON	TENMON	TENLOAI	SL_CONLAI	SOLUONG
1	1	PHỞ	MÓN NƯ...	50	50
2	2	BÚN CHÀ	MÓN NƯ...	50	50
3	3	BÚN NEM NUÔNG	MÓN NƯ...	50	50
4	4	CƠM CHIẾN	CƠM	50	50
5	5	BÁNH XÉO	BÁNH MĂN	50	50
6	6	BÁNH BÈO	BÁNH MĂN	50	50
7	7	BÁNH KHỌT	BÁNH MĂN	50	50
8	8	CHÈ BƯỜI	CHÈ	50	50
9	9	CHÈ HẠT SEN	CHÈ	50	50
10	10	CHÈ KHÚC BẠCH	CHÈ	50	50

Hình 2.1.b.2

iii. *Giải pháp:* Set cơ chế khóa Read Committed

Chạy giao tác 1 đến lệnh waitfor delay '00:00:10', sau đó chạy hết tác 2, lúc này T2 không đọc được dữ liệu mà T1 thay đổi do T1 vẫn giữ khóa X và chưa kết thúc giao tác.

Sau khi T1 rollback, T2 lập tức thu được dữ liệu ban đầu.

The screenshot shows two SQL Server Management Studio windows. The left window (T1) contains the following script:

```
--Dirty Read - Lai Gia Phu - 1712662
--T1
BEGIN TRAN
EXEC sp_CapNhatSoPhanAn 1,1,'2019-01-01',50,10
Waitfor delay '00:00:10'
ROLLBACK TRAN
```

The right window (T2) contains the following script:

```
--T2
BEGIN TRAN
set tran isolation level READ COMMITTED
EXEC sp_XemThucDon 1,'2019-01-01'
COMMIT TRAN
```

Below the scripts, the 'Messages' pane for T1 shows:

```
(1 row affected)
Sửa thành công
Completion time: 2020-01-01T19:23:20.7910167+07:00
```

The 'Results' pane for T2 displays a table of menu items:

MAMON	TENMON	TENLOAI	SL_CONLAI	SOLUONG
1	PHỞ	MÓN NƯ...	50	50
2	BÚN CHÀ	MÓN NƯ...	50	50
3	BÚN NEM NƯỚNG	MÓN NƯ...	50	50
4	CƠM CHIẾN	CƠM	50	50
5	BÁNH XÈO	BÁNH MÃ...	50	50
6	BÁNH BÈO	BÁNH MÃ...	50	50
7	BÁNH KHỌT	BÁNH MÃ...	50	50
8	CHÈ BƯỜI	CHÈ	50	50
9	CHÈ HẠT SEN	CHÈ	50	50
10	CHÈ KHÚC BẠCH	CHÈ	50	50

Hình 2.1.b.3

c. *Lỗi c*

❖ **Sinh viên thực hiện:** Nguyễn Đoàn Tấn Phúc – 1712671

i. *Kịch bản lỗi*

Giả sử nhân viên 1 thêm món vào thực đơn, nhân viên 2 xem thực đơn thì sẽ bị đọc lỗi nếu giao tác T1 không commit mà rollback lại. Ở ví dụ này, để xảy ra lỗi thì giả lập mức cô lập cho giao tác T2 là Read Uncommitted.

T1	T2
<pre>BEGIN TRAN EXEC sp_ThemMonThucDon 1,2,'2020-01-01',50,50 Waitfor delay '00:00:10'</pre>	<pre>BEGIN TRAN set tran isolation level READ UNCOMMITTED</pre>

ROLLBACK TRAN

EXEC sp_XemThucDon 1,'2020-01-

01'

COMMIT TRAN

ii. Chạy lỗi trên SQL

Đoạn test case của trường hợp Dirty Read:

Chạy giao tác 1 đến lệnh waitfor delay '00:00:10', sau đó chạy hết tác 2 và ta có kết quả:

The screenshot shows the Microsoft SQL Server Management Studio interface with two query panes and their results.

Object Explorer: Shows the database structure for 'DESKTOP-S378DD7' (SQL Server 12.0.0). It includes the 'HuongViet' database with various tables like CHINHANH, CHTIETDONHANG, DHTHUCUyen, DONHANG, GIO, LOAIMON, MONAN, NHANVIEN, THANHVIEN, and THUCDON.

Testcase02.sql: Contains the following script:

```
SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED
BEGIN TRAN
EXEC sp_XemThucDon 1,'2020-01-01'
COMMIT TRAN
```

Results pane: Shows the output of the first part of the script. A screenshot of the results grid shows a single row with columns MAMON, TENMON, TENLOAI, SL_CON, and SOLUONG. The values are MÓN NUỚC, 50, 50.

Testcase01.sql: Contains the following script:

```
--Dirty Read
--T1
BEGIN TRAN
set transaction isolation level READ UNCOMMITTED
EXEC sp_ThemMonThucDon 1,2,'2020-01-01',50,50
Waitfor delay '00:00:10'
ROLLBACK TRAN
--T2
BEGIN TRAN
set transaction isolation level READ COMMITTED
EXEC sp_XemThucDon 1,'2020-01-01'
COMMIT TRAN
```

Results pane: Shows the output of the second part of the script. A screenshot of the results grid shows a single row with columns MAMON, TENMON, TENLOAI, SL_CON, and SOLUONG. The values are BÚN CHÁ, MÓN NUỚC, 50, 50.

Hình 2.1.c.1

- ⇒ Giao tác 2 đã đọc dữ liệu mà giao tác 1 đã cập nhật nhưng chưa rollback với nội dung thêm 1 món vào thực đơn.

Sau khi Giao tác 1 rollback, thử truy vấn lại T2. Ta thấy kết quả đã không còn giống ban đầu.

```

--T1
BEGIN TRAN
set tran isolation level READ UNCOMMITTED
EXEC sp_XemThucDon 1,'2020-01-01'
COMMIT TRAN

--T2
BEGIN TRAN
set tran isolation level READ UNCOMMITTED
EXEC sp_XemThucDon 1,'2020-01-01'
COMMIT TRAN

--Sửa mức cờ lặp T2 thành READ COMMITTED
--T2
BEGIN TRAN
set tran isolation level READ COMMITTED
EXEC sp_XemThucDon 1,'2019-01-01'
COMMIT TRAN

```

Hình 2.1.c.2

iii. Giải pháp: Set cơ chế khóa Read Committed

Chạy giao tác 1 đến lệnh `waitfor delay '00:00:10'`, sau đó chạy hết tác 2, lúc này T2 không đọc được dữ liệu mà T1 thay đổi do T1 vẫn chưa kết thúc giao tác.

```

--T1
BEGIN TRAN
set tran isolation level READ COMMITTED
EXEC sp_XemThucDon 1,'2019-01-01'
COMMIT TRAN

--T2
BEGIN TRAN
set tran isolation level READ UNCOMMITTED
EXEC sp_XemThucDon 1,'2020-01-01'
COMMIT TRAN

--Sửa mức cờ lặp T2 thành READ COMMITTED
--T2
BEGIN TRAN
set tran isolation level READ COMMITTED
EXEC sp_XemThucDon 1,'2019-01-01'
COMMIT TRAN

```

Hình 2.1.c.3

Sau khi T1 rollback, T2 lập tức thu được dữ liệu ban đầu.

```

--T1
BEGIN TRAN
set tran isolation level READ COMMITTED
EXEC sp_ThemMonThucDon 1,'2020-01-01'
COMMIT TRAN

--T2
BEGIN TRAN
set tran isolation level READ UNCOMMITTED
EXEC sp_ThemMonThucDon 1,'2020-01-01'
COMMIT TRAN

```

Hình 2.1.c.4

d. Lỗi d

❖ **Sinh viên thực hiện:** Trịnh Đức Thanh – 1712769

i. Kịch bản lỗi

Quản lí hệ thống muốn cập nhật giá tiền món ăn 1 thành 25 000 nhưng cập nhật nhầm món ăn 2, khi đó có khách hàng online vào xem món ăn 2 thì thấy có giá mới là 25 000. Sau đó, quản lí phát hiện cập nhật nhầm và rollback, giá tiền của món 2 trở lại thành giá ban đầu. Ở đây, ta set cơ chế khóa Read Uncommitted cho giao tác T2.

T1	T2
<pre> BEGIN TRAN UPDATE MONAN SET GIATIEN=25000 WHERE MAMON=2 Waitfor delay '00:00:20' </pre>	<pre> BEGIN TRAN SET TRAN ISOLATION LEVEL READ UNCOMMITTED </pre>

ROLLBACK

SELECT * FROM MONAN

COMMIT

ii. Chạy lỗi trên SQL

Chạy đoạn test case gây lỗi như hình 2.1.d.1:

The screenshot shows two side-by-side SQL Server Management Studio windows. The left window, titled 'Test1.sql - DELL-P... (DELL\DELL (52))', contains the following script:

```
1 --Người thực hiện: Trịnh Đức Thành - 1712769
2
3 --Lỗi 1: Dirty Read
4 BEGIN TRAN
5 UPDATE MONAN
6 SET GIATIEN=25000
7 WHERE MAMON=2
8 Waitfor delay '00:00:20'
9 ROLLBACK
10
11
12
13
14
15
16
```

The right window, titled 'Test2.sql - DELL-P... (DELL\DELL (56))', contains the following script:

```
1 --Người thực hiện: Trịnh Đức Thành - 1712769
2
3 --Lỗi 1: Dirty Read
4 BEGIN TRAN
5 SET TRAN ISOLATION LEVEL READ UNCOMMITTED
6 SELECT * FROM MONAN
7 COMMIT
8
9 SELECT * FROM MONAN
```

Both windows show the 'Messages' tab with the message '(1 row(s) affected)' and 'Query executed successfully.' status.

The 'Results' tab in the right window displays a table of food items:

MAMON	TENMON	MALOAI	GIATIEN
1	PHỞ	1	50000
2	BÚN CHÁ	1	25000
3	BÚN NEM NUÔNG	1	40000
4	CƠM CHIÊN	2	30000
5	BÁNH XÈO	3	30000
6	BÁNH BÈO	3	25000
7	BÁNH KHÔT	3	20000
8	CHÈ BUỐI	4	20000
9	CHÈ HẠT SEN	4	20000
10	CHÈ KHÚC BẮCH	4	30000

Hình 2.1.d.1

Do set cơ chế khóa Read Uncommitted nên trong khoảng thời gian 20s, T2 đã chạy lệnh đọc mà không chờ T1 nhả khóa, nên dữ liệu đọc bị sai: Giá tiền món ăn 2 thành 25 000đ.

⇒ Gây ra lỗi Dirty Read.

Sau khi T1 rollback, thì T2 đọc lại dữ liệu thấy giá tiền của món ăn 2 đã trở lại thành 50 000đ như trước (hình 2.1.d.2).

```

Test1.sql - DELL-P... (DELL-PC\DELL (52))*
1 --Người thực hiện: Trịnh Đức Thành - 1712769
2
3 --Lỗi 1: Dirty Read
4 BEGIN TRAN
5 UPDATE MONAN
6 SET GIATIEN=25000
7 WHERE MAMON=2
8 Waitfor delay '00:00:20'
9 ROLLBACK
10
11
12
13
14
15
16

Messages
(1 row(s) affected)

100 % ▾
Query executed successfully. | DELL-PC (12.0 RTM) | DELL-PC\DELL (52) | HuongViet | 00:00:20 | 0 rows

Test2.sql - DELL-P... (DELL-PC\DELL (56))*
1 --Người thực hiện: Trịnh Đức Thành - 1712769
2
3 --Lỗi 1: Dirty Read
4 BEGIN TRAN
5 SET TRAN ISOLATION LEVEL READ UNCOMMITTED
6 SELECT * FROM MONAN
7 COMMIT
8
9
10
11
12
13
14
15
16

Results
MAMON TENMON MALOAI GIATIEN
1 1 PHỞ 1 50000
2 2 BÚN CHA 1 50000
3 3 BÚN NEM NUÔNG 1 40000
4 4 COM CHIÊN 2 30000
5 5 BÁNH XÈO 3 30000
6 6 BÁNH BÈO 3 25000
7 7 BÁNH KHỐT 3 20000
8 8 CHÈ BUỘI 4 20000
9 9 CHÈ HẠT SEN 4 20000
10 10 CHÈ KHÚC BẠCH 4 30000

Messages
Query executed successfully. | DELL-PC (12.0 RTM) | DELL-PC\DELL (56) | HuongViet | 00:00:00 | 10 rows

```

Hình 2.1.d.2

iii. Giải pháp: Set cơ chế khóa Read Committed.

T1	Khóa	T2	Khóa
		Set -> Read Committed	
BEGIN TRAN UPDATE MONAN SET GIATIEN=25000 WHERE MAMON=2 Waitfor delay '00:00:20'	T1: Xin khóa X SQL: Cấp khóa X		
		BEGIN TRAN SELECT * FROM MONAN	T2: Xin khóa S SQL: Không cấp khóa S do T1 đang giữ khóa X
ROLLBACK		COMMIT	

Ta set cơ chế khóa Read Committed cho giao tác T2, lúc này các câu truy vấn sẽ thực hiện theo thứ tự: Chạy truy vấn của T1 -> Chạy lệnh waitfor delay '00:00:20' -> Rollback T1 -> Chạy truy vấn của T2 -> Commit T2.

Khi đó, T2 không còn gặp lỗi Dirty Read nữa và đọc được các dữ liệu đúng (hình 2.1.d.3).

```

SQLQuery1.sql - DELL-PC\DELL (56)* > Test1.sql - DELL-P... (DELL-PC\DELL (52))
1  --Người thực hiện: Trịnh Đức Thành - 1712769
2
3  -Lỗi 1: Dirty Read
4  BEGIN TRAN
5  UPDATE MONAN
6  SET GIATIEN=25000
7  WHERE MAMON = 2
8  Waitfor delay '00:00:20'
9  ROLLBACK
10
11
12
13
14
15
16

(1 row(s) affected)

Item(s) Saved Ln 4 Col 1 Ch 1 INS

```



```

SQLQuery1.sql - DELL-PC\DELL (57)* > Test2.sql - DELL-P... (DELL-PC\DELL (55))
1  --Người thực hiện: Trịnh Đức Thành - 1712769
2
3  -Lỗi 1: Dirty Read
4  BEGIN TRAN
5  SET TRAN ISOLATION LEVEL READ COMMITTED
6  SELECT * FROM MONAN
7  COMMIT
8
9
10
11
12
13
14
15
16

Item(s) Saved Ln 4 Col 1 Ch 1 INS

```

Hình 2.1.d.3

e. Lỗi e

❖ **Sinh viên thực hiện:** Dương Khánh Vi – 1712899

i. Kịch bản lỗi

Giả sử khách hàng số 1 vào xem và thấy số lượng còn lại của món số 1 tại chi nhánh 1 là 1, người này tiến hành đặt hàng nhưng chưa commit thao tác. Trong lúc này thì khách hàng số 2 cùng vào xem món 1 tại chi nhánh 1 và thấy số lượng còn lại là 0, người này không đặt nữa. Khách 1 không đặt món nữa và rollback lại, lúc này khách hàng số 2 đã đọc dữ liệu sai. Ở ví dụ này, để xảy ra lỗi thì giả lập mức cô lập cho giao tác T2 là Read Uncommitted.

T1	T2
<pre> BEGIN TRAN UPDATE THUCDON SET SL_CONLAI = 0 WHERE MACN = 1 AND MAMON = 1 waitfor delay'00:00:10' </pre>	<pre> BEGIN TRAN </pre>

ROLLBACK	<pre>set tran isolation level READ UNCOMMITTED SELECT SL_CONLAI FROM THUCDON WHERE MATD = 1 AND MACN = 1 COMMIT</pre>
----------	--

ii. Chạy lỗi trên SQL

Đoạn test case của trường hợp Dirty Read:

```
SQLQuery3.sql - L_MJ28QOA\Asus (59)* X SQLQuery1.sql - L_MJ28QOA\Asus (55)*
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 1: Dirty Read
BEGIN TRAN
UPDATE THUCDON
SET SL_CONLAI = 0
WHERE MACN = 1 AND MAMON = 1
waitfor delay '00:00:10'
ROLLBACK

SQLQuery3.sql - L_MJ28QOA\Asus (60)* X SQLQuery1.sql - L_MJ28QOA\Asus (52)*
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 1: Dirty Read
BEGIN TRAN
set tran isolation level READ
UNCOMMITTED
SELECT SL_CONLAI FROM THUCDON WHERE MACN = 1 AND MAMON = 1
COMMIT
```

Hình 2.1.e.1

Chạy giao tác 1 đến lệnh waitfor delay '00:00:10', sau đó chạy hết tác 2 và ta có kết quả:

```
SQLQuery3.sql - L_MJ28QOA\Asus (59)* X SQLQuery1.sql - L_MJ28QOA\Asus (55)*
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 1: Dirty Read
BEGIN TRAN
UPDATE THUCDON
SET SL_CONLAI = 0
WHERE MACN = 1 AND MAMON = 1
waitfor delay '00:00:10'
ROLLBACK

SQLQuery3.sql - L_MJ28QOA\Asus (60)* X SQLQuery1.sql - L_MJ28QOA\Asus (52)*
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 1: Dirty Read
BEGIN TRAN
set tran isolation level READ
UNCOMMITTED
SELECT SL_CONLAI FROM THUCDON WHERE MACN = 1 AND MAMON = 1
COMMIT
```

Messages	Results
(1 row(s) affected)	SL_CONLAI 1 0

Hình 2.1.e.2

⇒ Giao tác 2 đã đọc dữ liệu mà giao tác 1 đã cập nhật nhưng chưa rollback với nội dung Số lượng còn lại của món ăn có mã 1 là 0.

Sau khi Giao tác 1 rollback, thử truy vấn lại thì số lượng còn lại đã trở lại 50.

```

SQLQuery4.sql - L...MJ28QOA\Asus (56)* X SQLQuery3.sql - L...MJ28QOA\Asus (59))*
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
SELECT SL_CONLAI FROM THUCDON WHERE MACN = 1 AND MAMON = 1

100 % < >
Results Messages
SL_CONLAI
1 50

Query exec... | LAPTOP-NMJ28QOA (12.0 SP1) | LAPTOP-NMJ28QOA\Asus (56) | HuongViet | 00:00:00 | 1 rows

```

Hình 2.1.e.3

iii. Giải pháp: Set cơ chế khóa Read Committed

T1	KHÓA	T2	KHÓA
	Set -> Read Committed		
UPDATE THUCDON SET SOLUONGCONLAI = 1 WHERE MATD = 1 AND MACN = 1 waitfor delay'00:00:10'	T1: xin X(THUCDON) SQL: cấp X(THUCDON) và giữ X cho đến hết giao tác		
		SELECT SOLUONGCONLAI FROM THUCDON WHERE MATD = 1 AND MACN = 1 Commit	T2: xin khóa S(THUCDON) SQL: Rỗng
Rollback			

Chạy giao tác 1 đến lệnh waitfor delay '00:00:10', sau đó chạy hết tác 2, lúc này T2 không đọc được dữ liệu mà T1 thay đổi do T1 vẫn chưa kết thúc giao tác.

```

SQLQuery3.sql - L...MJ28QOA\Asus (59)* X
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 1: Dirty Read
BEGIN TRAN
UPDATE THUCDON
SET SL_CONLAI = 0
WHERE MACN = 1 AND MAMON = 1
waitfor delay '00:00:10'
ROLLBACK

100 % < >
Messages
(1 row(s) affected)

100 % < >
Query exec... | LAPTOP-NMJ28QOA (12.0 SP1) | LAPTOP-NMJ28QOA\Asus (59) | HuongViet | 00:00:10 | 0 rows

SQLQuery3.sql - LAP...s (60) Executing... X stored của khách..MJ28QOA\Asus (61)*
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 1: Dirty Read
BEGIN TRAN
set tran isolation level READ COMMITTED
SELECT SL_CONLAI FROM THUCDON WHERE MACN = 1 AND MAMON = 1
COMMIT

100 % < >
Results Messages
SL_CONLAI
1 50

0 Executing q... | LAPTOP-NMJ28QOA (12.0 SP1) | LAPTOP-NMJ28QOA\Asus (60) | HuongViet | 00:00:13 | 0 rows

```

Hình 2.1.e.4

Sau khi T1 rollback, T2 lập tức thu được dữ liệu ban đầu.

```

SQLQuery3.sql - L...MJ28QOA\Asus (59)* X
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 1: Dirty Read
BEGIN TRAN
UPDATE THUCDON
SET SL_CONLAI = 0
WHERE MACN = 1 AND MAMON = 1
waitfor delay '00:00:10'
ROLLBACK

100 % < >
Messages
Command(s) completed successfully.

100 % < >
Query exec... | LAPTOP-NMJ28QOA (12.0 SP1) | LAPTOP-NMJ28QOA\Asus (59) | HuongViet | 00:00:00 | 0 rows

SQLQuery3.sql - L...MJ28QOA\Asus (60)* X stored của khách..MJ28QOA\Asus (61)*
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 1: Dirty Read
BEGIN TRAN
set tran isolation level READ COMMITTED
SELECT SL_CONLAI FROM THUCDON WHERE MACN = 1 AND MAMON = 1
COMMIT

100 % < >
Results Messages
SL_CONLAI
1 50

0 Executing q... | LAPTOP-NMJ28QOA (12.0 SP1) | LAPTOP-NMJ28QOA\Asus (60) | HuongViet | 00:02:11 | 1 rows

```

Hình 2.1.e.5

2.2. Unrepeatable Read

a. Lỗi a

❖ **Sinh viên thực hiện:** Nguyễn Thị Ngọc Hân – 1712415

i. Kịch bản lỗi

Khách hàng đang xem giá số lượng món ăn số 1 ở chi nhánh 1, có khách hàng khác đặt món. Quản lí tiến hành cập nhật số lượng thì khách hàng đang xem sẽ thấy số lượng thay đổi.

T1	T2
BEGIN TRAN	

```

SELECT SL_CONLAI FROM THUCDON
WHERE MAMON = 1 AND MACN =
1 AND NGAY='2019-11-30'

```

```
waitfor delay'00:00:15'
```

```
BEGIN TRAN
```

```
UPDATE THUCDON SET SL_CONLAI=
150 WHERE MAMON = 1 AND
MACN = 1 AND NGAY='2019-11-30'
```

```
COMMIT
```

```

SELECT SL_CONLAI FROM THUCDON
WHERE MAMON = 1 AND MACN =
1 AND NGAY='2019-11-30'

```

```
COMMIT
```

ii. Chạy lỗi trên SQL

Đoạn test case của trường hợp Unrepeatable Read: (Giả lập lấy cơ chế Read Committed)

```

SQLQuery1.sql - D...09AIR6\Admin (51)* | SQLQuery3.sql - D...09AIR6\Admin (54)*
1
2
3 --THIẾC HIỆN: NGUYỄN THỊ NGỌC HÂN
4 --MSSV: 1712415
5 BEGIN TRAN
6 SELECT SL_CONLAI FROM THUCDON WHERE MAMON = 1 AND MACN = 1 AND NGAY='2019-11-30'
7 waitfor delay'00:00:15'
8
9
10 SELECT SL_CONLAI FROM THUCDON WHERE MAMON = 1 AND MACN = 1 AND NGAY='2019-11-30'
11 COMMIT
12
13
14
15
16
17 ROLLBACK
18

SQLQuery1.sql - D...09AIR6\Admin (51)* | SQLQuery3.sql - D...09AIR6\Admin (54)*
1
2 --THIẾC HIỆN: NGUYỄN THỊ NGỌC HÂN
3 --MSSV: 1712415
4 BEGIN TRAN
5 UPDATE THUCDON SET SL_CONLAI= 150
6 WHERE MAMON = 1 AND MACN = 1 AND NGAY='2019-11-30'
7 COMMIT
8
9

```

Hình 2.2.a.1

Chạy giao tác 1 đến lệnh waitfor delay '00:00:15', sau đó chạy hết giao tác 2 và ta có kết quả:

```

SQLQuery1.sql - D...09AIR6\Admin (51)* X SQLQuery3.sql - D...09AIR6\Admin (54)*
1
2
3 --THỰC HIỆN: NGUYỄN THỊ NGỌC HÂN
4 |-MSSV: 1712415
5 BEGIN TRAN
6 SELECT SL_CONLAI FROM THUDON WHERE MAMON = 1 AND MACN = 1 AND NGAY='2019-11-30'
7 waitfor delay'00:00:15'
8
9
10 SELECT SL_CONLAI FROM THUDON WHERE MAMON = 1 AND MACN = 1 AND NGAY='2019-11-30'
11 COMMIT
12
13
14
15
16
17 ROLLBACK
18

100 %
Results Messages
SL_CONLAI
1 76

SQLQuery1.sql - D...09AIR6\Admin (51)* X SQLQuery3.sql - D...09AIR6\Admin (54)* X
1
2 --THỰC HIỆN: NGUYỄN THỊ NGỌC HÂN
3 |-MSSV: 1712415
4 BEGIN TRAN
5 UPDATE THUDON SET SL_CONLAI= 150
6 WHERE MAMON = 1 AND MACN = 1 AND NGAY='2019-11-30'
7 COMMIT
8
9

100 %
Messages
(1 row(s) affected)

```

Hình 2.2.a.2

Giao tác 1 đọc dữ liệu của mã tv = 1 lần 2:

```

SQLQuery1.sql - D...09AIR6\Admin (51)* X SQLQuery3.sql - D...09AIR6\Admin (54)*
1
2
3 --THỰC HIỆN: NGUYỄN THỊ NGỌC HÂN
4 |-MSSV: 1712415
5 BEGIN TRAN
6 SELECT SL_CONLAI FROM THUDON WHERE MAMON = 1 AND MACN = 1 AND NGAY='2019-11-30'
7 waitfor delay'00:00:15'
8
9
10 SELECT SL_CONLAI FROM THUDON WHERE MAMON = 1 AND MACN = 1 AND NGAY='2019-11-30'
11 COMMIT
12
13
14
15
16
17 ROLLBACK
18

100 %
Results Messages
SL_CONLAI
1 150

```

Hình 2.2.a.3

- ⇒ Ở lần đọc thứ 2 này, Giao tác 1 đã thấy dữ liệu có sự thay đổi, cụ thể là số lượng sản phẩm đã bị thay đổi.
- iii. *Giải pháp:* Set cơ chế khóa Repeatable Read

T1	KHOÁ	T2	KHOÁ
Set REPEATABLE READ			
SELECT SL_CONLAI FROM THUCDON WHERE MAMON = 1 AND MACN = 1 AND NGAY='2019-11-30' waitfor delay '00:00:15'	T1: xin khóa S(THUCDON) SQL cấp S(THANHVIEN) và giữ đến cuối giao tác		
		UPDATE THUCDON SET SL_CONLAI= 150 WHERE MAMON = 1 AND MACN = 1 AND NGAY='2019-11-30'	T2 xin khóa X(THUCDON) SQL: rỗng
		COMMIT	
SELECT SL_CONLAI FROM THUCDON WHERE MAMON = 1 AND MACN = 1 AND NGAY='2019-11-30'			
COMMIT			

Chạy giao tác 1 đến lệnh waitfor delay '00:00:15', sau đó chạy hết giao tác 2, lúc này T2 không cập nhật được dữ liệu do T1 set mức cô lập Repeatable Read. Khóa S đã được giữ cho đến hết giao tác, T2 muốn xin khóa X thì phải đợi T1 hoàn thành xong giao tác.

```

SQLQuery1.sql - D...09AIR6\Admin (51)* × SQLQuery3.sql - DES...n (54)* Executing...
1 --THỰC HIỆN: NGUYỄN THỊ NGỌC HÂN
2 --MSSV: 1712415
3 BEGIN TRAN
4 SET TRAN ISOLATION level repeatable read
5 SELECT SL_CONLAI FROM THUCDON WHERE MAMON = 1 AND MACN = 1 AND NGAY='2019-11-30'
6 waitfor delay '00:00:15'
7
8
9
10
11 SELECT SL_CONLAI FROM THUCDON WHERE MAMON = 1 AND MACN = 1 AND NGAY='2019-11-30'
12 COMMIT
13
14
15
16
17
18 ROLLBACK
100 % ▾
Messages
SL_CONLAI
1 76

SQLQuery1.sql - D...09AIR6\Admin (51)* × SQLQuery3.sql - D...09AIR6\Admin (54)* ×
1 --THỰC HIỆN: NGUYỄN THỊ NGỌC HÂN
2 --MSSV: 1712415
3 BEGIN TRAN
4 UPDATE THUCDON SET SL_CONLAI= 150
5 WHERE MAMON = 1 AND MACN = 1 AND NGAY='2019-11-30'
6 COMMIT
7
8
9
100 % ▾
Messages

```

Hình 2.2.a.4

Sau khi kết thúc giao tác T1, T2 mới vào thực hiện cập nhật dữ liệu của MATV = 1, và đây là kết quả sau khi cập nhật:

```

1 --THỰC HIỆN: NGUYỄN THỊ NGỌC HÂN
2 |-MSSV: 1712415
3
4 BEGIN TRAN
5 UPDATE THUDON SET SL_CONLAI= 150
6 WHERE MAMON = 1 AND MACN = 1 AND NGAY='2019-11-30'
7
8 COMMIT
9
10
11
12
13
14
15
16

```

100 % < Messages
(1 row(s) affected)

```

1 --THỰC HIỆN: NGUYỄN THỊ NGỌC HÂN
2 |-MSSV: 1712415
3
4 BEGIN TRAN
5 SET TRAN ISOLATION level repeatable read
6 SELECT SL_CONLAI FROM THUDON WHERE MAMON = 1 AND MACN = 1 AND NGAY='2019-11-30'
7 waitfor delay '00:00:15'
8
9
10
11
12
13
14
15
16

```

100 % < Results Messages
SL_CONLAI
1 150

Hình 2.2.a.5

b. Lỗi b

❖ **Sinh viên thực hiện:** Lai Gia Phú – 1712662

i. Kịch bản lỗi

Khách hàng 1 thực hiện xem thực đơn của chi nhánh 1, thì thấy số lượng còn lại của món 1 là 50. Sau đó có khách hàng 2 tới đặt 5 phần món 1 ở chi nhánh 1 và được nhân viên lễ tân tiếp nhận đơn. Khách hàng 1 thực hiện xem thực đơn lại một lần nữa thì thấy số lượng còn lại của món 1 là 45.

T1	T2
<pre>BEGIN TRAN EXEC sp_XemThucDon 1,'2020-01-01' Waitfor delay '00:00:10' EXEC sp_XemThucDon 1,'2020-01-01'</pre>	<pre>BEGIN TRAN EXEC sp_InDH null,null,1,N'Trực tiếp',N'Qua thẻ' EXEC sp_InCTDH 1,1,5 COMMIT TRAN</pre>

ii. Chạy lỗi trên SQL

Đoạn test case của trường hợp Unrepeatable Read: (Giả lập lấy cơ chế Read Committed)

Chạy giao tác 1 đến lệnh waitfor delay '00:00:10', sau đó chạy hết giao tác 2 và ta có kết quả:

The screenshot shows two SSMS windows side-by-side.

Left Window (TestCase02.sql):

```
--T1
BEGIN TRAN
EXEC sp_XemThucDon 1,'2020-01-01'
Waitfor delay '00:00:10'
EXEC sp_XemThucDon 1,'2020-01-01'
COMMIT TRAN
```

Right Window (SQLQuery3.sql):

```
--T2
BEGIN TRAN
EXEC sp_InDH null,null,1,N'Trực tiếp',N'Qua thẻ'
EXEC sp_InCTDH 1,1,5
COMMIT TRAN
```

Results pane (Left):

MAMON	TENMON	TENLOAI	SL_CONLAI	SOLUONG
1	PHỞ	MÓN NƯỚC	50	50

Messages pane (Left):

(1 row affected)

Results pane (Right):

MAMON	TENMON	TENLOAI	SL_CONLAI	SOLUONG
1	PHỞ	MÓN NƯỚC	45	50

Messages pane (Right):

(1 row affected)

(1 row affected)

(1 row affected)

Completion time: 2020-01-01T10:22:29.0137212+07:00

Hình 2.2.b.1

- ⇒ Giao tác 1 tiến hành xem thực đơn chi nhánh 1 kết quả giống hệt dữ liệu ban đầu(chưa chỉnh sửa).

Giao tác 1 xem thực đơn chi nhánh 1 lần 2:

- ⇒ Ở lần đọc thứ 2 này, Giao tác 1 đã thấy dữ liệu có sự thay đổi, cụ thể là số lượng còn lại chỉ còn 45.

iii. Giải pháp: Set cơ chế khóa Repeatable Read

Chạy giao tác 1 đến lệnh waitfor delay '00:00:10', sau đó chạy hết giao tác 2, lúc này T2 không cập nhật được dữ liệu do T1 set mức độ lặp Repeatable Read. Khóa S đã được giữ cho đến hết giao tác, T2 muốn xin khóa X thì phải đợi T1 hoàn thành xong giao tác. Khi T1 vừa hoàn thành thì T2 mới cập nhật số lượng còn lại.

```

--T1
BEGIN TRAN
set tran isolation level Repeatable Read
EXEC sp_XemThucDon 1,'2020-01-01'
Waitfor delay '00:00:10'
EXEC sp_XemThucDon 1,'2020-01-01'
COMMIT TRAN

--T2
BEGIN TRAN
EXEC sp_InDH null,null,1,N'Tryc tiếp',N'Qua the'
EXEC sp_InCTDH 1,1,5
COMMIT TRAN

```

Results pane (T1):

MAMON	TENMON	TENLOAI	SL_CONLAI	SOLUONG
1	Phở	MÓN NƯỚNG	50	50

Results pane (T2):

MAMON	TENMON	TENLOAI	SL_CONLAI	SOLUONG
1	Phở	MÓN NƯỚNG	50	50

Messages pane (T1):

- (1 row affected)
- (1 row affected)
- (1 row affected)
- (1 row affected)

Completion time: 2020-01-01T10:42:24.9287429+07:00

Messages pane (T2):

- (1 row affected)
- (1 row affected)
- (1 row affected)
- (1 row affected)

Hình 2.2.b.2

c. *Lỗi c*

❖ **Sinh viên thực hiện:** Nguyễn Đoàn Tấn Phúc – 1712671

i. *Kịch bản lỗi*

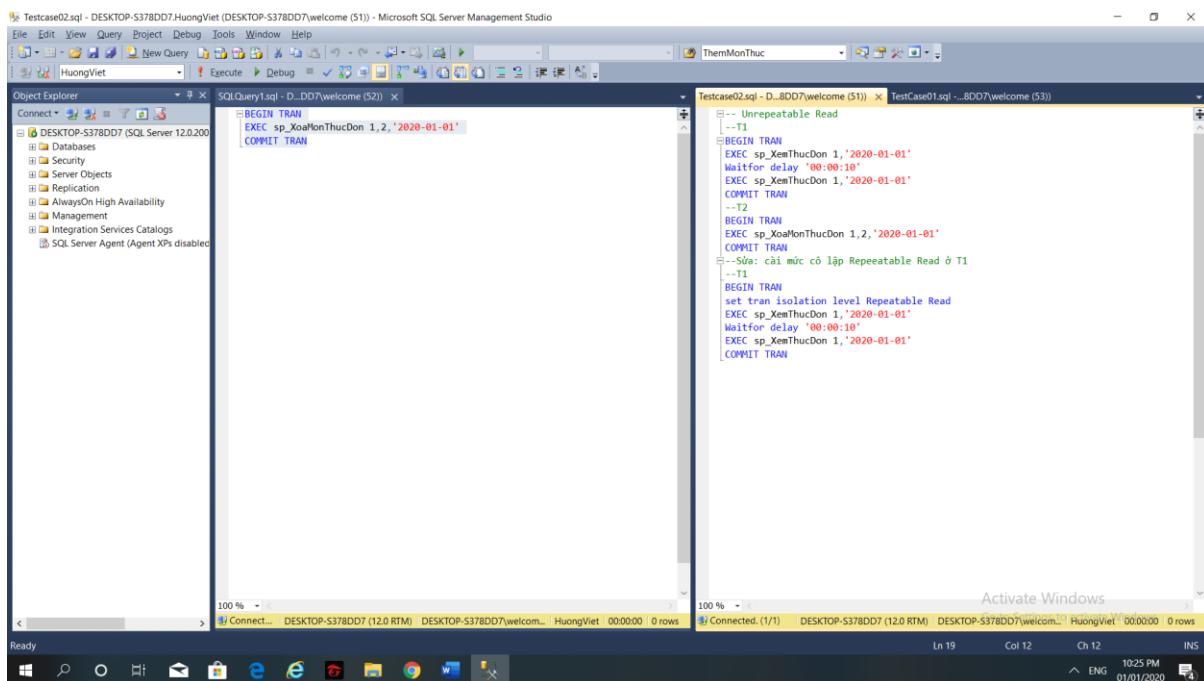
T1 xem trong thực đơn, sau đó T2 xóa trong phần T1 đã xem sau đó T1 xem lại thì kết quả không giống nhau.

T1	T2
<pre> BEGIN TRAN EXEC sp_XemThucDon 1,'2020-01- 01' Waitfor delay '00:00:10' </pre>	<pre> BEGIN TRAN EXEC sp_XoaMonThucDon 1,2,'2020- 01-01' COMMIT TRAN </pre>

<pre>EXEC sp_XemThucDon 1,'2020-01- 01' COMMIT TRAN</pre>	
--	--

ii. Chạy lỗi trên SQL

Đoạn test case của trường hợp Unrepeatable Read: (Giả lập lấy cơ chế Read Committed)



```
Testcase02.sql - DESKTOP-S378DD7.HuongViet (DESKTOP-S378DD7\welcome (51)) - Microsoft SQL Server Management Studio
File Edit View Query Project Debug Tools Window Help
HuongViet Execute Debug
Object Explorer
Connect SQL Server Object Explorer
DESKTOP-S378DD7 (SQL Server 12.0.200
Databases Security Server Objects Replication AlwaysOn High Availability Management Integration Services Catalogs SQL Server Agent (Agent XPs disabled)
SQLQuery1.sql - D...DD7\welcome (52) Testcase02.sql - D...DD7\welcome (51) TestCase01.sql - D...DD7\welcome (53)
BEGIN TRAN
EXEC sp_XemThucDon 1,2,'2020-01-01'
COMMIT TRAN
-- Unrepeatable Read
--T1
BEGIN TRAN
EXEC sp_XemThucDon 1,'2020-01-01'
Waitfor delay '00:00:10'
EXEC sp_XemThucDon 1,'2020-01-01'
COMMIT TRAN
--T2
BEGIN TRAN
EXEC sp_XemThucDon 1,2,'2020-01-01'
COMMIT TRAN
--Sửa: cần mức độ lặp Repeatable Read ở T1
T1
BEGIN TRAN
set tran isolation level Repeatable Read
EXEC sp_XemThucDon 1,'2020-01-01'
Waitfor delay '00:00:10'
EXEC sp_XemThucDon 1,'2020-01-01'
COMMIT TRAN
Activate Windows
Ln 19 Col 12 Ch 12 INS
^ ENG 10:25 PM 01/01/2020
```

Hình 2.2.c.1

Chạy giao tác 1 đến lệnh waitfor delay '00:00:10', sau đó chạy hết tác 2 và ta có kết quả:

The screenshot shows the Microsoft SQL Server Management Studio interface with two query panes and a results pane.

- Object Explorer:** Shows the database structure for DESKTOP-S378DD7 (SQL Server 12.0.200).
- Query1.sql:** Contains a transaction script:


```

BEGIN TRAN
EXEC sp_XoaMonThucDon 1,2,'2020-01-01'
COMMIT TRAN
      
```

 The message pane indicates "Xóa thành công" (Deleted successfully) with 1 row affected.
- Testcase02.sql:** Contains a transaction script:


```

--T1
BEGIN TRAN
EXEC sp_XemThucDon 1,'2020-01-01'
Waitfor delay '00:00:10'
EXEC sp_XemThucDon 1,'2020-01-01'
COMMIT TRAN
--T2
BEGIN TRAN
EXEC sp_XoaMonThucDon 1,2,'2020-01-01'
COMMIT TRAN
      
```

 A note in the code states "--Sửa: cài mức cô lập Repeatable Read ở T1". The results pane shows the state of the table after both transactions have run:

MAMON	TENMON	TENLOAI	SL_CON...	SOLUO...
2	BÚN CHẢ	MÓN NƯỚC	50	50

Hình 2.2.c.2

- ⇒ Giao tác 1 tiến hành đọc dữ liệu của thực đơn và kết quả giống hệt dữ liệu ban đầu(chưa xóa).

Giao tác 1 đọc dữ liệu lần 2.

- ⇒ Ở lần đọc thứ 2 này, Giao tác 1 đã thấy dữ liệu có sự thay đổi, cụ thể là dữ liệu đã bị xóa

iii. Giải pháp: Set cơ chế khóa Repeatable Read

Chạy giao tác 1 đến lệnh waitfor delay '00:00:10', sau đó chạy hết giao tác 2, lúc này T2 không xóa được dữ liệu do T1 set mức cô lập Repeatable Read.

Khóa S đã được giữ cho đến hết giao tác, T2 muốn xin khóa X thì phải đợi T1 hoàn thành xong giao tác.

```

-- Testcase01.sql - DESKTOP-S378DD7.HuongViet (DESKTOP-S378DD7\welcome (52)) - Microsoft SQL Server Management Studio
File Edit View Query Project Debug Tools Window Help
Connect New Query Execute Debug
Object Explorer
Connect Databases Security Server Objects Replication AlwaysOn High Availability Management Integration Services Catalogs SQL Server Agent (Agent XPs disabled)
DESKTOP-S378DD7 (SQL Server 12.0.2000.18)
-- Testcase01.sql - DESKTOP-S378DD7\welcome (52) x
BEGIN TRAN
EXEC sp_XoaMonThucDon 1,2,'2020-01-01'
COMMIT TRAN
-- Testcase02.sql - DESKTOP-S378DD7.HuongViet (DESKTOP-S378DD7\welcome (51)) x TestCase01.sql -...8DD7\welcome (53)
EXEC sp_XemThucDon 1,'2020-01-01'
Waitfor delay '00:00:10'
EXEC sp_XemThucDon 1,'2020-01-01'
COMMIT TRAN
--T2
BEGIN TRAN
EXEC sp_XoaMonThucDon 1,2,'2020-01-01'
COMMIT TRAN
--Sửa: cài mức độ lặp Repeatable Read ở T1
--T1
BEGIN TRAN
set tran isolation level Repeatable Read
EXEC sp_XemThucDon 1,'2020-01-01'
Waitfor delay '00:00:10'
EXEC sp_XemThucDon 1,'2020-01-01'
COMMIT TRAN

```

Messages

(1 row(s) affected)
Xóa thành công

Results

MAMON	TENMON	TENLOAI	SL_CON...	SOLUO...
1	BÚN CHẢ	MÓN NƯỚC	50	50

Activate Windows

Query executed on DESKTOP-S378DD7 (12.0 RTM) DESKTOP-S378DD7\welcome (51) by HuongViet at 00:00:08 0 rows

Ln 1 Col 1 Ch 1 INS

Ready

Hình 2.2.c.3

Sau khi kết thúc giao tác T1, T2 mới vào thực hiện xóa dữ liệu, và đây là kết quả sau khi cập nhật:

```

-- Testcase01.sql - DESKTOP-S378DD7.HuongViet (DESKTOP-S378DD7\welcome (51)) - Microsoft SQL Server Management Studio
File Edit View Query Project Debug Tools Window Help
Connect New Query Execute Debug
Object Explorer
Connect Databases Security Server Objects Replication AlwaysOn High Availability Management Integration Services Catalogs SQL Server Agent (Agent XPs disabled)
DESKTOP-S378DD7 (SQL Server 12.0.2000.18)
-- Testcase01.sql - DESKTOP-S378DD7\welcome (52) x
BEGIN TRAN
EXEC sp_XoaMonThucDon 1,2,'2020-01-01'
COMMIT TRAN
-- Testcase02.sql - DESKTOP-S378DD7.HuongViet (DESKTOP-S378DD7\welcome (51)) x TestCase01.sql -...8DD7\welcome (53)
EXEC sp_XemThucDon 1,'2020-01-01'
Waitfor delay '00:00:10'
EXEC sp_XemThucDon 1,'2020-01-01'
COMMIT TRAN
--T2
BEGIN TRAN
EXEC sp_XoaMonThucDon 1,2,'2020-01-01'
COMMIT TRAN
--Sửa: cài mức độ lặp Repeatable Read ở T1
--T1
BEGIN TRAN
set tran isolation level Repeatable Read
EXEC sp_XemThucDon 1,'2020-01-01'
Waitfor delay '00:00:10'
EXEC sp_XemThucDon 1,'2020-01-01'
COMMIT TRAN

```

Messages

(1 row(s) affected)
Xóa thành công

Results

MAMON	TENMON	TENLOAI	SL_CON...	SOLUO...

Activate Windows

Query executed on DESKTOP-S378DD7 (12.0 RTM) DESKTOP-S378DD7\welcome (51) by HuongViet at 00:00:10 0 rows

Ln 14 Col 1 Ch 1 INS

Ready

Hình 2.2.c.4

d. Lỗi d

❖ **Sinh viên thực hiện:** Trịnh Đức Thanh – 1712769

i. Kịch bản lỗi

Khách hàng 1 đang muốn đặt món ăn 1 ở chi nhánh 1 với số lượng là 40, khi họ vào xem thực đơn thì thấy món ăn ban đầu là 50 và dự định tiến hành đặt hàng, nhưng có có khách hàng 2 đặt hàng nhanh hơn, và số lượng còn lại của món ăn 1 thay đổi thành 30. Khi khách hàng 1 xem lại thì thấy số lượng đã bị thay đổi và không đủ để cung cấp.

T1	T2
<pre>BEGIN TRAN SELECT SOLUONGCONLAI FROM THUCDON WHERE MAMON = 1 AND MACN = 1 waitfor delay'00:00:20'</pre>	<pre>BEGIN TRAN UPDATE THUCDON SET SOLUONGCONLAI = 30 WHERE MAMON = 1 AND MACN = 1 COMMIT SELECT SOLUONGCONLAI FROM THUCDON WHERE MAMON = 1 AND MACN = 1 COMMIT</pre>

ii. Chạy lỗi trên SQL

Chạy đoạn test case gây lỗi như hình 2.2.d.1:

```

--Lỗi 2: Unrepeatable
BEGIN TRAN
SELECT SL_CONLAI
FROM THUCDON
WHERE MANON = 1 AND MACN = 1
waitfor delay '00:00:20'

SELECT SL_CONLAI
FROM THUCDON
WHERE MANON = 1 AND MACN = 1
COMMIT

```

Results

SL_CONLAI
50

(1 row(s) affected)

```

--Lỗi 2: Unrepeatable
BEGIN TRAN
UPDATE THUCDON
SET SL_CONLAI = 30
WHERE MANON = 1 AND MACN = 1
COMMIT

```

Results

(1 row(s) affected)

Hình 2.2.d.1

Do cơ chế khóa mặc định là Read Committed nên khi T1 đang giữ khóa S, thì T2 vẫn xin được khóa X. Trong khoảng thời gian 20s, T2 đã chạy truy vấn và thực hiện cập nhật số lượng còn lại của món ăn 1 thành 30 và nhả khóa (commit).

T1 thực hiện tiếp truy vấn đọc thì thấy dữ liệu đã bị thay đổi: Số lượng còn lại của món ăn 1 từ 50 thành 30 (Hình 2.2.d.2).

⇒ Gây ra lỗi UnrepeatableRead.

```

--Lỗi 2: Unrepeatable
BEGIN TRAN
SELECT SL_CONLAI
FROM THUCDON
WHERE MANON = 1 AND MACN = 1
waitfor delay '00:00:20'

SELECT SL_CONLAI
FROM THUCDON
WHERE MANON = 1 AND MACN = 1
COMMIT

```

Results

SL_CONLAI
50

(1 row(s) affected)

```

--Lỗi 2: Unrepeatable
BEGIN TRAN
UPDATE THUCDON
SET SL_CONLAI = 30
WHERE MANON = 1 AND MACN = 1
COMMIT

```

Results

(1 row(s) affected)

Hình 2.2.d.2

iii. Giải pháp: Set cơ chế khóa Repeatable Read

T1	Khóa	T2	Khóa
Set -> Repeatable Read			
BEGIN TRAN SELECT SOLUONGCONLAI FROM THUCDON WHERE MAMON = 1 AND MACN = 1 waitfor delay'00:00:20'	T1: Xin khóa S SQL: Cấp khóa S T1: Giữ khóa S đến hết giao tác T1	BEGIN TRAN UPDATE THUCDON SET SOLUONGCONLAI = 30 WHERE MAMON = 1 AND MACN = 1	T2: Xin khóa X SQL: Không cấp khóa X do T1 đang giữ khóa S
		COMMIT	
SELECT SOLUONGCONLAI FROM THUCDON WHERE MAMON = 1 AND MACN = 1 COMMIT			

Ta set cơ chế khóa Repeatable Read cho giao tác T1, lúc này T1 sẽ được giữ khóa S cho đến hết giao tác, và T2 không được cấp khóa X cho đến khi giao tác T1 hoàn thành. Các câu truy vấn sẽ thực hiện theo thứ tự: Chạy truy vấn thứ 1 của T1 -> Chạy lệnh waitfor delay '00:00:20' -> Tiếp tục chạy truy vấn thứ 2 của T1 -> Commit T1 -> Chạy truy vấn của T2 -> Commit T2.

Khi đó, T1 không còn gặp lỗi UnrepeatableRead nữa và đọc được các dữ liệu đúng (Hình 2.2.d.3).

```

SQLQuery1.sql - DELL-PC\DELL (56)* > Test1.sql - DELL-P... (DELL-PC\DELL (52))
17 --Người thực hiện: Trịnh Đức Thành - 1712769
18
19 --Lỗi 2: Unrepeatable
20 BEGIN TRAN
21 SET TRAN ISOLATION LEVEL REPEATABLE READ
22 SELECT SL_CONLAI
23 FROM THUCDON
24 WHERE MAMON = 1 AND MACN = 1
25 waitfor delay'00:00:20'
26
27 SELECT SL_CONLAI
28 FROM THUCDON
29 WHERE MAMON = 1 AND MACN = 1
30 COMMIT
31
32

100 % < > Results Messages
SL_CONLAI
1 50

< > Query executed successfully. DELL-PC (12.0 RTM) | DELL-PC\DELL (56) | HuongViet | 00:00:20 | 1 rows
Item(s) Saved Ln 20 Col 1 Ch 1 INS . . .
```



```

SQLQuery1.sql - DELL-PC\DELL (57) Executing... > Test2.sql - DELL-P... (DELL-PC\DELL (55))
16
17 --Người thực hiện: Trịnh Đức Thành - 1712769
18
19 --Lỗi 2: Unrepeatable
20 BEGIN TRAN
21 UPDATE THUCDON
22 SET SL_CONLAI = 30
23 WHERE MAMON = 1 AND MACN = 1
24 COMMIT
25
26
27
28
29
30
31

100 % < > Results Messages
< > Executing query... DELL-PC (12.0 RTM) | DELL-PC\DELL (57) | HuongViet | 00:00:19 | 0 rows
Item(s) Saved Ln 20 Col 1 Ch 1 INS . . .
```

Hình 2.2.d.3

e. Lỗi e

❖ **Sinh viên thực hiện:** Dương Khánh Vi – 1712899

i. Kịch bản lỗi

Khách hàng có mã thành viên = 1 đang xem thông tin tài khoản của mình ở chi nhánh 1, thì có người khác cũng vào tài khoản này và sửa thông tin DIACHI . Khách hàng thực hiện xem thông tin lại một lần nữa thì thông tin ban đầu đã bị thay đổi.

T1	T2
<pre>BEGIN TRAN SELECT * FROM THANHVIEN WHERE MATV = 1 waitfor delay'00:00:10'</pre>	<pre>BEGIN TRAN UPDATE THANHVIEN SET DIACHI = N'390 CAO THĂNG, PHƯỜNG 12, QUẬN 10' WHERE MATV = 1 COMMIT</pre>

ii. Chạy lỗi trên SQL

Đoạn test case của trường hợp Unrepeatable Read: (Giả lập lấy cơ chế Read Committed)

```

SQLQuery4.sql - L...MJ28QOA\Asus (56)* SQLQuery3.sql - L...MJ28QOA\Asus (59)*
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 2: Unrepeatable Read
BEGIN TRAN
SELECT * FROM THANHVIEN WHERE MATV = 1
waitfor delay '00:00:05'
SELECT * FROM THANHVIEN WHERE MATV = 1
COMMIT
GO

SQLQuery3.sql - L...MJ28QOA\Asus (60)* DATA DEMO TRANS...28QOA\Asus (51)
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 2: Unrepeatable read
BEGIN TRAN
UPDATE THANHVIEN
SET DIACHI = N'390 CAO THÁNG, PHƯỜNG 12, QUẬN 10' WHERE MATV = 1
COMMIT
GO

```

Hình 2.2.e.1

Chạy giao tác 1 đến lệnh waitfor delay '00:00:10', sau đó chạy hết tác 2 và ta có kết quả:

MATV	TENTV	MATKHAU	CMND	NGAYSINH	DIACHI	CHINHANHDK	EMAIL
1	AAA	1	321456984	2000-03-17	112 Nguyễn Văn Cừ, Phường 6, Quận 5	1	AAA

(1 row(s) affected)									
---------------------	--	--	--	--	--	--	--	--	--

Hình 2.2.e.2

⇒ Giao tác 1 tiến hành đọc dữ liệu của MATV = 1 và kết quả giống hệt dữ liệu ban đầu(chưa chỉnh sửa).

Giao tác 1 đọc dữ liệu của MATV = 1 lần 2:

MATV	TENTV	MATKHAU	CMND	NGAYSINH	DIACHI	CHINHANHDK	EMAIL	DIEMTICHLUY	LOAITHIE
1	AAA	1	321456984	2000-03-17	390 CAO THÁNG, PHƯỜNG 12, QUẬN 10	1	AAA@GMAIL.COM	NULL	SILVER

Hình 2.2.e.3

⇒ Ở lần đọc thứ 2 này, Giao tác 1 đã thấy dữ liệu có sự thay đổi, cụ thể là địa chỉ đã bị thay đổi.

iii. Giải pháp: Set cơ chế khóa Repeatable Read

T1	Khóa	T2	Khóa
Set -> Repeatable Read			
SELECT * FROM THANHVIEN WHERE MATV = 1 waitfor delay'00:00:10'	<u>T1:</u> xin S(TANHVIEN) <u>SQL:</u> cấp xin S(TANHVIEN) và giữ S cho hết giao tác		
		UPDATE THANHVIEN SET DIACHI = N'390 CAO THẮNG, PHƯỜNG 12, QUẬN 10' WHERE MATV = 1	<u>T2:</u> xin X(TANHVIEN) <u>SQL:</u> Rỗng
SELECT * FROM THANHVIEN WHERE MATV = 1			
COMMIT		COMMIT	

Chạy giao tác 1 đến lệnh waitfor delay '00:00:15', sau đó chạy hết giao tác 2, lúc này T2 không cập nhật được dữ liệu do T1 set mức độ lặp Repeatable Read. Khóa S đã được giữ cho đến hết giao tác, T2 muốn xin khóa X thì phải đợi T1 hoàn thành xong giao tác.

The screenshot shows two separate sessions in SQL Server Management Studio. Both sessions are connected to the database 'L...MJ28QOA\Asus'.

Session 1 (Left):

```

/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 2: Unrepeatable read
BEGIN TRAN
set tran isolation level REPEATABLE READ
SELECT * FROM THANHVIEEN WHERE MATV = 1
waitfor delay '00:00:15'
SELECT * FROM THANHVIEEN WHERE MATV = 1
COMMIT
  
```

Session 2 (Right):

```

/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 2: Unrepeatable read
BEGIN TRAN
UPDATE THANHVIEEN
SET DIACHI = N'390 CAO THÁNG, PHƯỜNG 12, QUẬN 10' WHERE MATV = 1
COMMIT
  
```

Both sessions show the same results:

MATV	TENTV	MATKHAU	CMND	NGAYSINH	DIACHI	CHINHANHDK
1	AAA	1	321456984	2000-03-17	122, Nguyễn Văn Cừ, phường 10, quận 5, TPHCM	1

Hình 2.2.e.4

Sau khi kết thúc giao tác T1, T2 mới vào thực hiện cập nhật dữ liệu của MATV = 1, và đây là kết quả sau khi cập nhật:

The screenshot shows two sessions in SQL Server Management Studio. Both sessions are connected to the database 'L...MJ28QOA\Asus'.

Session 1 (Left):

```

/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 2: Unrepeatable read
BEGIN TRAN
UPDATE THANHVIEEN
SET DIACHI = N'390 CAO THÁNG, PHƯỜNG 12, QUẬN 10' WHERE MATV = 1
COMMIT
  
```

Output: (1 row(s) affected)

Session 2 (Right):

```

/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 2: Unrepeatable read
BEGIN TRAN
UPDATE THANHVIEEN
SET DIACHI = N'390 CAO THÁNG, PHƯỜNG 12, QUẬN 10' WHERE MATV = 1
COMMIT
SELECT * FROM THANHVIEEN WHERE MATV = 1
  
```

Output: Results

MATV	TENTV	MATKHAU	CMND	NGAYSINH	DIACHI	CHINHANHDK	EMAIL	DIEMTICHLUY	LOAITHIEU
1	AAA	1	321456984	2000-03-17	390 CAO THÁNG, PHƯỜNG 12, QUẬN 10	1	AAA@GMAIL.COM	NULL	SILVER

Hình 2.2.e.5

2.3. Phantom

a. Lỗi a

❖ **Sinh viên thực hiện:** Nguyễn Thị Ngọc Hân – 1712415

i. Kịch bản lỗi

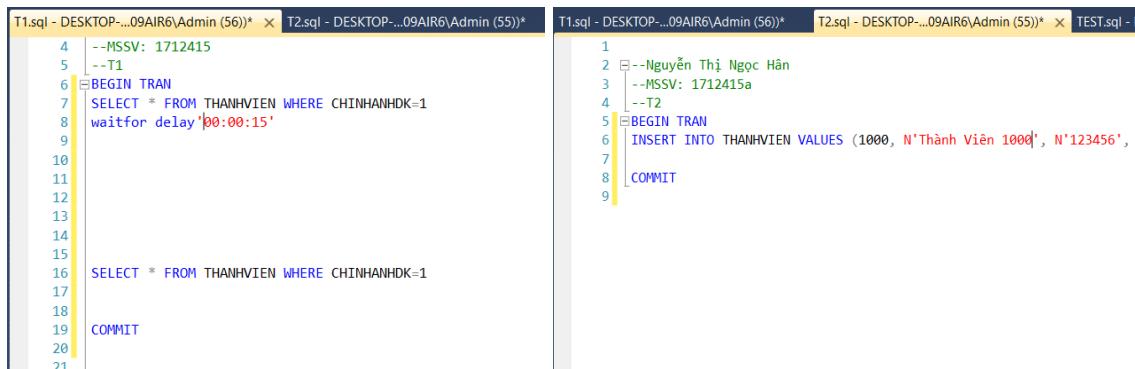
Quản lí chi nhánh thống kê những thành viên của mình, tại thời điểm đó thì nhân viên lễ tân lại chèn thêm 1 thành viên mới.



BEGIN TRAN SELECT * FROM THANHVIEN WHERE CHINHANHDK=1 waitfor delay'00:00:15'	BEGIN TRAN INSERT INTO THANHVIEN VALUES (...) COMMIT
SELECT * FROM THANHVIEN WHERE CHINHANHDK=1 COMMIT	

ii. Chạy lỗi trên SQL

Đoạn test case của trường hợp Phantom: (Giả lập lấy cơ chế Read Committed)



```

T1.sql - DESKTOP-...09AIR6\Admin (56)* T2.sql - DESKTOP-...09AIR6\Admin (55)* TEST.sql - DESKTOP-...09AIR6\Admin (55)*
1 --MSSV: 1712415
2 --T1
3 BEGIN TRAN
4 SELECT * FROM THANHVIEN WHERE CHINHANHDK=1
5 waitfor delay '00:00:15'
6
7
8
9
10
11
12
13
14
15
16 SELECT * FROM THANHVIEN WHERE CHINHANHDK=1
17
18
19 COMMIT
20
21

1 --Nguyễn Thị Ngọc Hân
2 --MSSV: 1712415a
3 --T2
4 BEGIN TRAN
5 INSERT INTO THANHVIEN VALUES (1000, N'Thanh Viên 1000', N'123456',
6
7
8 COMMIT
9

```

Hình 2.3.a.1

Chạy giao tác 1 đến lệnh waitfor delay '00:00:15', sau đó chạy hết tác 2 và ta có kết quả:



```

T1.sql - DESKTOP-...09AIR6\Admin (56)* T2.sql - DESKTOP-...09AIR6\Admin (55)* TEST.sql - DESKTO...
1 --Nguyễn Thị Ngọc Hân
2 --MSSV: 1712415a
3 --T2
4 BEGIN TRAN
5 INSERT INTO THANHVIEN VALUES (1000, N'Thanh Viên 1000', N'123456', N'32190')
6
7
8 COMMIT
9

100 % < Messages
(1 row(s) affected)

```

```

1 USE [DESKTOP-B09AIR6]
2 GO
3 SET NOCOUNT ON;
4 BEGIN TRAN
5 SELECT * FROM THANHVIEN WHERE CHINHANHDK=1
6 waitfor delay'00:00:15'
7
8 COMMIT
9
10
11
12
13
14
15
16
17
18
19
20
21

```

Results

MATV	TENTV	MATKH...	CMND	NGAYSINH	DIACHI	CHINHANHDK	EMAIL	DIEMTICHLUY	LOAITHE
1	Nguyễn Văn E	1	354677122	1998-10-12	309 Lê Hồng Phong, Phường 11, Quận 10	1	DDD@gmail.com	NULL	SILVER
2	Thành Viên 2	123456	321706811	2000-10-12	Phường 3, Quận 3 TP.HCM	1	thanhvien2@gmail.com	30000000	Dimond
3	Thành Viên 3	123456	321706812	2000-10-12	Phường 4, Quận 4 TP.HCM	1	thanhvien3@gmail.com	30000000	Dimond
4	Thành Viên 4	123456	321706813	2000-10-12	Phường 5, Quận 5 TP.HCM	1	thanhvien4@gmail.com	30000000	Dimond
5	Thành Viên 5	123456	321706814	2000-10-12	Phường 6, Quận 6 TP.HCM	1	thanhvien5@gmail.com	30000000	Dimond
6	Thành Viên 6	123456	321706815	2000-10-12	Phường 7, Quận 7 TP.HCM	1	thanhvien6@gmail.com	30000000	Dimond
7	Thành Viên 7	123456	321706816	2000-10-12	Phường 8, Quận 8 TP.HCM	1	thanhvien7@gmail.com	30000000	Dimond
8	Thành Viên 8	123456	321706817	2000-10-12	Phường 9, Quận 9 TP.HCM	1	thanhvien8@gmail.com	30000000	Dimond
9	Thành Viên 9	123456	321706818	2000-10-12	Phường 10, Quận 10 TP.HCM	1	thanhvien9@gmail.com	30000000	Dimond
10	Thành Viên 10	123456	321706819	2000-10-12	Phường 11, Quận 11 TP.HCM	1	thanhvien10@gmail.com	30000000	Dimond
11	Thành Viên 11	123456	321706820	2000-10-12	Phường 12, Quận 12 TP.HCM	1	thanhvien11@gmail.com	30000000	Dimond
12	Thành Viên 12	123456	321706821	2000-10-12	Phường 13, Quận 1 TP.HCM	1	thanhvien12@gmail.com	30000000	Dimond
13	Thành Viên 13	123456	321706822	2000-10-12	Phường 14, Quận 2 TP.HCM	1	thanhvien13@gmail.com	30000000	Dimond
14	Thành Viên 14	123456	321706823	2000-10-12	Phường 15, Quận 3 TP.HCM	1	thanhvien14@gmail.com	30000000	Dimond
15	Thành Viên 15	123456	321706824	2000-10-12	Phường 16, Quận 4 TP.HCM	1	thanhvien15@gmail.com	30000000	Dimond

Query executed successfully.

Hình 2.3.a.2

⇒ Giao tác 1 xem Thành viên của chi nhánh 1 và nhận được dữ liệu ban đầu (không cập nhật/ thêm). Giao tác 2 đã thêm dữ liệu thành công.

Giao tác 1 tiến hành xem thành viên của chi nhánh 1 một lần nữa:

```

1
2
3 --Nguyễn Thị Ngọc Hân
4 --MSSV: 1712415
5 --T1
6 BEGIN TRAN
7 SELECT * FROM THANHVIEN WHERE CHINHANHDK=1
8 waitfor delay'00:00:15'
9
10
11
12
13
14 SELECT * FROM THANHVIEN WHERE CHINHANHDK=1
15
16
17 COMMIT
18

```

Results

MATV	TENTV	MATKH...	CMND	NGAYSINH	DIACHI	CHINHANHDK	EMAIL	DIEMTICHLUY	LOAITHE
1	Nguyễn Văn E	1	354677122	1998-10-12	309 Lê Hồng Phong, Phường 11, Quận 10	1	DDD@gmail.com	NULL	SILVER
2	Thành Viên 2	123456	321706811	2000-10-12	Phường 3, Quận 3 TP.HCM	1	thanhvien2@gmail.com	30000000	Dimond
3	Thành Viên 3	123456	321706812	2000-10-12	Phường 4, Quận 4 TP.HCM	1	thanhvien3@gmail.com	30000000	Dimond
4	Thành Viên 4	123456	321706813	2000-10-12	Phường 5, Quận 5 TP.HCM	1	thanhvien4@gmail.com	30000000	Dimond
5	Thành Viên 5	123456	321706814	2000-10-12	Phường 6, Quận 6 TP.HCM	1	thanhvien5@gmail.com	30000000	Dimond
6	Thành Viên 6	123456	321706815	2000-10-12	Phường 7, Quận 7 TP.HCM	1	thanhvien6@gmail.com	30000000	Dimond
7	Thành Viên 7	123456	321706816	2000-10-12	Phường 8, Quận 8 TP.HCM	1	thanhvien7@gmail.com	30000000	Dimond
8	Thành Viên 8	123456	321706817	2000-10-12	Phường 9, Quận 9 TP.HCM	1	thanhvien8@gmail.com	30000000	Dimond
9	Thành Viên 9	123456	321706818	2000-10-12	Phường 10, Quận 10 TP.HCM	1	thanhvien9@gmail.com	30000000	Dimond
10	Thành Viên 10	123456	321706819	2000-10-12	Phường 11, Quận 11 TP.HCM	1	thanhvien10@gmail.com	30000000	Dimond
11	Thành Viên 11	123456	321706820	2000-10-12	Phường 12, Quận 12 TP.HCM	1	thanhvien11@gmail.com	30000000	Dimond
12	Thành Viên 12	123456	321706821	2000-10-12	Phường 13, Quận 1 TP.HCM	1	thanhvien12@gmail.com	30000000	Dimond
13	Thành Viên 13	123456	321706822	2000-10-12	Phường 14, Quận 2 TP.HCM	1	thanhvien13@gmail.com	30000000	Dimond
14	Thành Viên 14	123456	321706823	2000-10-12	Phường 15, Quận 3 TP.HCM	1	thanhvien14@gmail.com	30000000	Dimond
15	Thành Viên 15	123456	321706824	2000-10-12	Phường 16, Quận 4 TP.HCM	1	thanhvien15@gmail.com	30000000	Dimond

Query executed successfully.

Hình 2.3.a.3

⇒ Giao tác 1 đọc dữ liệu và thấy kết quả bị thêm 1 dòng.

iii. Giải pháp: Set cơ chế khóa Serializable

T1	KHOÁ	T2	KHOÁ
SET SERIALIZABLE		SET committed	
BEGIN TRAN SELECT * FROM THANHVIEN WHERE CHINHANHDK=1 waitfor delay'00:00:15'	T1 xin khoá S(TANHVIEN) SQL: cấp S(TANHVIEN) cho T1 và giữ cho đến hết giao tác		
		BEGIN TRAN INSERT INTO THANHVIEN VALUES (...)	T2: xin khoá X(TANHVIEN) SQL: rỗng
SELECT * FROM THANHVIEN WHERE CHINHANHDK=1			
COMMIT		COMMIT	

Chạy giao tác 1 đến lệnh waitfor delay '00:00:15', sau đó chạy hết giao tác 2, lúc này T2 không thêm được dữ liệu do T1 set mức độ cô lập Serializable. Khóa S đã được giữ cho đến hết giao tác, không cho insert dòng thỏa điều kiện thiết lập khóa, nên T2 muốn thêm dữ liệu thì phải đợi T1 commit.

```

T1.sql - DESKTOP-B0...09AIR6\Admin (56)* TEST.sql - DESKTOP-B0...n (55) Executing...* TEST.sql - DESKTOP-B0...n (55) Executing...* TEST.sql - DESKTOP-B0...n (55) Executing...*
1 --Nguyễn Thị Ngọc Hân
2
3
4 --T2
5 BEGIN TRAN
6 INSERT INTO THANHVIEN VALUES (1002, N'Thành Viên 1000', N'123456', N'32
7
8 COMMIT
9

```

MÃV	HỌ TÊN	MÃTKH	DOB	NGAYTHMH	DIACHI	CHINHANHDK	EMAIL	DEMKHCLL	CATHER
1	Nguyễn Văn A	123456789	1998-10-12	2001-10-10	300 Lô Hồng Phong, Phường 11, Quận 10	1	123@gmail.com	NULL	SILVER
2	Thành Viên 2	123456	2000-10-10	2004-10-10	Phường 3, Quận 3 TP.HCM	1	thanhtuan2@gmail.com	30000000	Diamond
3	Thành Viên 3	123456	2001-09-01	2004-10-10	Phường 4, Quận 1 TP.HCM	1	thanhtuan3@gmail.com	30000000	Diamond
4	Thành Viên 4	123456	2001-09-01	2004-10-10	Phường 5, Quận 5 TP.HCM	1	thanhtuan4@gmail.com	30000000	Diamond
5	Thành Viên 5	123456	2000-10-12	2004-10-10	Phường 6, Quận 8 TP.HCM	1	thanhtuan5@gmail.com	30000000	Diamond
6	Thành Viên 6	123456	2000-10-12	2004-10-10	Phường 7, Quận 7 TP.HCM	1	thanhtuan6@gmail.com	30000000	Diamond
7	Thành Viên 7	123456	2000-10-12	2004-10-10	Phường 8, Quận 8 TP.HCM	1	thanhtuan7@gmail.com	30000000	Diamond
8	Thành Viên 8	123456	2000-10-12	2004-10-10	Phường 9, Quận 9 TP.HCM	1	thanhtuan8@gmail.com	30000000	Diamond
9	Thành Viên 9	123456	2000-10-12	2004-10-10	Phường 10, Quận 10 TP.HCM	1	thanhtuan9@gmail.com	30000000	Diamond
10	Thành Viên 10	123456	2000-10-12	2004-10-10	Phường 11, Quận 11 TP.HCM	1	thanhtuan10@gmail.com	30000000	Diamond
11	Thành Viên 11	123456	2000-10-12	2004-10-10	Phường 12, Quận 12 TP.HCM	1	thanhtuan11@gmail.com	30000000	Diamond
12	Thành Viên 12	123456	2000-10-12	2004-10-10	Phường 13, Quận 1 TP.HCM	1	thanhtuan12@gmail.com	30000000	Diamond
13	Thành Viên 13	123456	2000-10-12	2004-10-10	Phường 14, Quận 2 TP.HCM	1	thanhtuan13@gmail.com	30000000	Diamond
14	Thành Viên 14	123456	2000-10-12	2004-10-10	Phường 15, Quận 5 TP.HCM	1	thanhtuan14@gmail.com	30000000	Diamond

Hình 2.3.a.4

Đây là kết quả sau khi T1 commit:

The screenshot shows a SQL Server Management Studio window with three tabs: T1.sql, T2.sql, and TEST.sql. The T2.sql tab is active, displaying the following script:

```
1 --Nguyễn Thị Ngọc Hân
2 --MSSV: 1712415a
3 --T2
4 BEGIN TRAN
5 INSERT INTO THANHVIEN VALUES (1002, N'Thành Viên 1000', N'123456', N'321907810', CAST(N'1970-11-09' AS Date), N'Phường 10, Quận
6 100 % < Messages
7 COMMIT
8
9
```

The Messages pane shows the output: (1 row(s) affected).

The Results pane shows the message: Query executed successfully.

Hình 2.3.a.5

SELECT lại bảng thành viên ta được kết quả:

The screenshot shows a SQL Server Management Studio window with three tabs: T1.sql, T2.sql, and TEST.sql. The T2.sql tab is active, displaying the following script:

```
1
2
3 --Nguyễn Thị Ngọc Hân
4 --MSSV: 1712415
5 --T1
6 BEGIN TRAN
7 SET TRAN ISOLATION LEVEL SERIALIZABLE
8 SELECT * FROM THANHVIEN WHERE CHINHANHDK=1
9 waitfor delay '00:00:15'
10
11
12
13
14
15
16
17
18 COMMIT
```

The Results pane displays the following table:

MATV	TENTV	MATKHM	CMND	NGAYSINH	DIACHI	CHINHANHDK	EMAIL	DIEMTICHLUY	LOAITHE
1	Nguyễn VĂN E	1	354677122	1998-10-12	309 Lê Hồng Phong, Phường 11, Quận 10	1	DDD@gmail.com	NULL	SILVER
2	Thành Viên 2	123456	321706811	2000-10-12	Phường 3, Quận 3 TP.HCM	1	thanhvien2@gmail.com	3000000	Dimond
3	Thành Viên 3	123456	321706812	2000-10-12	Phường 4, Quận 4 TP.HCM	1	thanhvien3@gmail.com	3000000	Dimond
4	Thành Viên 4	123456	321706813	2000-10-12	Phường 5, Quận 5 TP.HCM	1	thanhvien4@gmail.com	3000000	Dimond
5	Thành Viên 5	123456	321706814	2000-10-12	Phường 6, Quận 6 TP.HCM	1	thanhvien5@gmail.com	3000000	Dimond
6	Thành Viên 6	123456	321706815	2000-10-12	Phường 7, Quận 7 TP.HCM	1	thanhvien6@gmail.com	3000000	Dimond
7	Thành Viên 7	123456	321706816	2000-10-12	Phường 8, Quận 8 TP.HCM	1	thanhvien7@gmail.com	3000000	Dimond
8	Thành Viên 8	123456	321706817	2000-10-12	Phường 9, Quận 9 TP.HCM	1	thanhvien8@gmail.com	3000000	Dimond
9	Thành Viên 9	123456	321706818	2000-10-12	Phường 10, Quận 10 TP.HCM	1	thanhvien9@gmail.com	3000000	Dimond
10	Thành Viên 10	123456	321706819	2000-10-12	Phường 11, Quận 11 TP.HCM	1	thanhvien10@gmail.com	3000000	Dimond
11	Thành Viên 11	123456	321706820	2000-10-12	Phường 12, Quận 12 TP.HCM	1	thanhvien11@gmail.com	3000000	Dimond
12	Thành Viên 12	123456	321706821	2000-10-12	Phường 13, Quận 1 TP.HCM	1	thanhvien12@gmail.com	3000000	Dimond
13	Thành Viên 13	123456	321706822	2000-10-12	Phường 14, Quận 2 TP.HCM	1	thanhvien13@gmail.com	3000000	Dimond
14	Thành Viên 14	123456	321706823	2000-10-12	Phường 15, Quận 3 TP.HCM	1	thanhvien14@gmail.com	3000000	Dimond
15	Thành Viên 15	123456	321706824	2000-10-12	Phường 16, Quận 4 TP.HCM	1	thanhvien15@gmail.com	3000000	Dimond

The Messages pane shows the message: Query executed successfully.

Hình 2.3.a.6

- ⇒ T2 thêm dữ liệu thành công
- b. Lỗi b
- ❖ **Sinh viên thực hiện:** Lai Gia Phú – 1712662
 - i. *Kịch bản lỗi*

Giả sử khách hàng xem thực đơn của chi nhánh 1 thấy chỉ có món 1. Sau đó, quản lý công ty có thêm món 2 vào thực đơn. Khi khách hàng xem lại thực đơn thì thấy xuất hiện thêm món 2.

T1	T2
<pre>BEGIN TRAN EXEC sp_XemThucDon 1,'2020-01-01' Waitfor delay '00:00:10'</pre>	<pre>BEGIN TRAN EXEC sp_ThemMonThucDon 1,2,'2020-01-01',50,50 COMMIT TRAN</pre>

ii. *Chạy lỗi trên SQL*

Đoạn test case của trường hợp Phantom: (Giả lập lấy cơ chế Read Committed)

Giao tác 1 xem thực đơn của chi nhánh 1 và nhận được dữ liệu ban đầu (không cập nhật/ thêm). Giao tác 2 đã thêm dữ liệu thành công.

Giao tác 1 tiến hành xem thực đơn của chi nhánh 1 một lần nữa:

```

--Phantom - Lai Gia Phu - 1712662
--T1
BEGIN TRAN
EXEC sp_XemThucDon 1, '2020-01-01'
Waitfor delay '00:00:10'
EXEC sp_XemThucDon 1, '2020-01-01'
COMMIT TRAN

--T2
BEGIN TRAN
EXEC sp_ThemMonThucDon 1, 2, '2020-01-01', 50, 50
COMMIT TRAN

```

Results

MAMON	TENMON	TENLOAI	SL_CONLAI	SOLUONG
1	PHỞ	MÓN NƯ...	50	50

Messages

(1 row affected)

Completion time: 2020-01-01T20:19:35.3521402+07:00

Results

MAMON	TENMON	TENLOAI	SL_CONLAI	SOLUONG
1	PHỞ	MÓN NƯ...	50	50
2	BÚN C...	MÓN NƯ...	50	50

Messages

Hình 2.3.b.1

⇒ Giao tác 1 đọc dữ liệu và thấy thêm 1 món

iii. Giải pháp: Set cơ chế khóa Serializable

Chạy giao tác 1 đến lệnh waitfor delay '00:00:10', sau đó chạy hết giao tác 2, lúc này T2 không thêm được dữ liệu do T1 set mức độ cô lập Serializable. Khóa S đã được giữ cho đến hết giao tác, không cho insert dòng thỏa điều kiện thiết lập khóa, nên T2 muốn thêm dữ liệu thì phải đợi T1 commit.

```

--Phantom - Lai Gia Phu - 1712662
--T1
BEGIN TRAN
set tran isolation level SERIALIZABLE
EXEC sp_XemThucDon 1, '2020-01-01'
Waitfor delay '00:00:10'
EXEC sp_XemThucDon 1, '2020-01-01'
COMMIT TRAN

--T2
BEGIN TRAN
EXEC sp_ThemMonThucDon 1, 2, '2020-01-01', 50, 50
COMMIT TRAN

```

Results

MAMON	TENMON	TENLOAI	SL_CONLAI	SOLUONG
1	PHỞ	MÓN NƯ...	50	50

Messages

(1 row affected)

Completion time: 2020-01-01T20:37:26.6302418+07:00

Results

MAMON	TENMON	TENLOAI	SL_CONLAI	SOLUONG
1	PHỞ	MÓN NƯ...	50	50
2	BÚN C...	MÓN NƯ...	50	50

Messages

Hình 2.3.b.2

Sau khi T2 commit

The screenshot shows the SQL Server Management Studio interface. In the top pane, there are two queries:

```
EXEC sp_XemThucDon 1, '2020-01-01'  
Waitfor delay '00:00:10'  
EXEC sp_XemThucDon 1, '2020-01-01'  
COMMIT TRAN
```

In the bottom pane, the 'Results' tab is selected, displaying the output of the second query:

	MAMON	TENMON	TENLOAI	SL_CONLAI	SOLUONG
1	1	PHỞ	MÓN NƯ...	50	50
2	2	BÚN C...	MÓN NƯ...	50	50

Hình 2.3.b.3

c. Lỗi c

❖ **Sinh viên thực hiện:** Nguyễn Đoàn Tấn Phúc – 1712671

i. Kích bản lỗi

Giả sử T1 thống kê vào ngày hôm nay tại chi nhánh 1. Sau đó T2 insert vào bảng đơn hàng. T1 thống kê lại thì thấy thêm dữ liệu T2 vừa thêm vào dù chưa commit. Lỗi phantom.

T1	T2
<pre>BEGIN TRAN EXEC sp_ThongKeTheoNgayCN 1 Waitfor delay '00:00:10'</pre>	<pre>BEGIN TRAN EXEC sp_InDH null,null,1,N'Trực tiếp',N'Qua thẻ' EXEC sp_InCTDH 1,1,1 COMMIT TRAN</pre>

EXEC sp_ThongKeTheoNgayCN 1

COMMIT TRAN

ii. Chạy lỗi trên SQL

Đoạn test case của trường hợp Phantom: (Giả lập lấy cơ chế Read Committed)

```
SQLQuery3.sql - L...M\J28QOA\Asus (59)* SQLQuery1.sql - L...M\J28QOA\Asus (55)*
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 4: Phantom
BEGIN TRAN
SELECT * FROM THANHVIEEN WHERE CHINHANHDK = 2
waitfor delay '00:00:10'
SELECT * FROM THANHVIEEN WHERE CHINHANHDK = 2
COMMIT

SQLQuery3.sql - L...M\J28QOA\Asus (60)* DATA DEMO TRANS...28QOA\Asus (51)*
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 4: Phantom
BEGIN TRAN
INSERT INTO THANHVIEEN
VALUES (10,N'ABC','1',3212532,'1997-07-07',NULL,
2,N'ABC@GMAIL.COM',NULL,N'SILVER')
COMMIT
```

Hình 2.3.c.1

Chạy giao tác 1 đến lệnh waitfor delay '00:00:10', sau đó chạy hết tác 2 và ta có kết quả:

⇒ Giao tác 1 xem dữ liệu của chi nhánh 2 và nhận được dữ liệu ban đầu (không cập nhật/ thêm). Giao tác 2 đã thêm dữ liệu thành công.

Giao tác 1 tiến hành xem dữ liệu của chi nhánh 1 một lần nữa:

```
Testcase04.sql - DESKTOP-S378DD7.HuongViet (DESKTOP-S378DD7\welcome (54)) - Microsoft SQL Server Management Studio
File Edit View Project Debug Tools Window Help
HuongViet
Object Explorer
DESKTOP-S378DD7 (SQL Server 12.0)
Databases
System Databases
Database Snapshots
HuongViet
Tables
System Tables
FileTables
dbo.CHINHANH
dbo.CHTIETDONHANG
dbo.DHTSTRUCTUYEN
dbo.DONHANG
dbo.GIO
dbo.LOAIMON
dbo.MONAN
dbo.NHANVIEN
dbo.THANHVIEEN
dbo.THUCDON
Views
Synonyms
Programmability
Service Broker
Storage
Full Text Catalogs
Partition Schemes
Partition Functions
Full Text Stopplists
Search Property Lists
Security
Partition_Vinabook
SaleManagement_Index
SaleManagement_NolIndex
TopiC4
VINABOOK

SQLQuery1.sql - D...DD7\welcome (53)* store.sql - DESKT...8DD7\welcome (51)
BEGIN TRAN
EXEC sp_InDH null,null,1,N'Trực tiếp',N'Qua thẻ'
EXEC sp_InCTOH 1,1,1
COMMIT TRAN
EXEC sp_ThemMonThuocDon 1,1,'2020-01-02',1,1

Messages
(1 row(s) affected)
(1 row(s) affected)
(1 row(s) affected)

Testcase04.sql - D...8DD7\welcome (54)
--Phantom
|---T1
| BEGIN TRAN
| EXEC sp_ThongKeTheoNgayCN 1
| Waitfor delay '00:00:10'
| EXEC sp_ThongKeTheoNgayCN 1
| COMMIT TRAN
|---T2
| BEGIN TRAN
| EXEC sp_InDH null,null,1,N'Trực tiếp',N'Qua thẻ'
| EXEC sp_InCTOH 1,1,5
| COMMIT TRAN
|---Sửa: mức cờ lặp Serializable ở T1
|---T1
| BEGIN TRAN
| set tran isolation level SERIALIZABLE
| EXEC sp_ThongKeTheoNgayCN 1
| Waitfor delay '00:00:10'
| EXEC sp_ThongKeTheoNgayCN 1
| COMMIT TRAN
Results Messages
NGAY DOANH_T...
1 2020-01-02 1

NGAY DOANH_T...
1 2020-01-02 50001

Activate Windows
Ready
Windows Taskbar
```

Hình 2.3.c.2

⇒ Giao tác 1 đọc dữ liệu và thấy kết quả khác.

iii. Giải pháp: Set cơ chế khóa Serializable

Chạy giao tác 1 đến lệnh waitfor delay '00:00:10', sau đó chạy hết giao tác 2, lúc này T2 không thêm được dữ liệu do T1 set mức độ khóa Serializable. Khóa S đã được giữ cho đến hết giao tác, không cho insert dòng thỏa điều kiện thiết lập khóa, nên T2 muốn thêm dữ liệu thì phải đợi T1 commit.

```
Testcase04.sql - DESKTOP-S378DD7.HuongViet (DESKTOP-S378DD7\welcome (52)) - Microsoft SQL Server Management Studio

Object Explorer
File Edit View Query Project Debug Tools Window Help
HuongViet
SQLQuery1.sql - D...8DD7\welcome (51) x
BEGIN TRAN
set tran isolation level SERIALIZABLE
EXEC sp_ThongKeTheoNgayCN 1
Waitfor delay '00:00:10'
EXEC sp_ThongKeTheoNgayCN 1
COMMIT TRAN
--T2
BEGIN TRAN
EXEC sp_InDHT null,null,1,N'Trục tiếp',N'Qua thẻ'
EXEC sp_InTDH 1,1,5
COMMIT TRAN
--Sửa: mức độ khóa Serializable ở T1
--T1
BEGIN TRAN
set tran isolation level SERIALIZABLE
EXEC sp_ThongKeTheoNgayCN 1
Waitfor delay '00:00:10'
EXEC sp_ThongKeTheoNgayCN 1
COMMIT TRAN

Results Messages
NGAY DOANH_T...
(1 rows) affected
(1 rows) affected
(1 rows) affected
(1 rows) affected

NGAY DOANH_T...
(1 rows) affected
(1 rows) affected
(1 rows) affected
(1 rows) affected

Activate Windows
Ln 8 Col 1 Ch 1 INS
^ ENG 3:50 PM 02/01/2020

Ready
Query ex... DESKTOP-S378DD7 (12.0 RTM) DESKTOP-S378DD7\welcome... HuongViet 00:00:10 0 rows
Query executed s... DESKTOP-S378DD7 (12.0 RTM) DESKTOP-S378DD7\welcome... HuongViet 00:00:08 0 rows
Ln 8 Col 1 Ch 1 INS
^ ENG 3:50 PM 02/01/2020
```

Hình 2.3.c.3

Đây là kết quả sau khi T1 commit:

```
Testcase04.sql - DESKTOP-S378DD7.HuongViet (DESKTOP-S378DD7\welcome (51)) - Microsoft SQL Server Management Studio

Object Explorer
File Edit View Query Project Debug Tools Window Help
HuongViet
SQLQuery1.sql - D...8DD7\welcome (51) x
Testcase04.sql - D...8DD7\welcome (52) x
BEGIN TRAN
set tran isolation level SERIALIZABLE
EXEC sp_ThongKeTheoNgayCN 1
Waitfor delay '00:00:10'
EXEC sp_ThongKeTheoNgayCN 1
COMMIT TRAN
--T2
BEGIN TRAN
EXEC sp_InDHT null,null,1,N'Trục tiếp',N'Qua thẻ'
EXEC sp_InTDH 1,1,5
COMMIT TRAN
--Sửa: mức độ khóa Serializable ở T1
--T1
BEGIN TRAN
set tran isolation level SERIALIZABLE
EXEC sp_ThongKeTheoNgayCN 1
Waitfor delay '00:00:10'
EXEC sp_ThongKeTheoNgayCN 1
COMMIT TRAN

Results Messages
NGAY DOANH_T...
1 2020-01-02 | 250000
(1 rows) affected
(1 rows) affected
(1 rows) affected
(1 rows) affected

Activate Windows
Ln 5 Col 1 Ch 1 INS
^ ENG 3:51 PM 02/01/2020

Ready
Query ex... DESKTOP-S378DD7 (12.0 RTM) DESKTOP-S378DD7\welcome... HuongViet 00:00:00 1 rows
Query executed s... DESKTOP-S378DD7 (12.0 RTM) DESKTOP-S378DD7\welcome... HuongViet 00:00:08 0 rows
Ln 5 Col 1 Ch 1 INS
^ ENG 3:51 PM 02/01/2020
```

Hình 2.3.c.4

- ⇒ T2 thêm dữ liệu thành công.
- d. Lỗi d
- ❖ **Sinh viên thực hiện:** Trịnh Đức Thanh – 1712769
 - i. **Kịch bản lỗi**

Nhân viên quản lý chi nhánh 1 đang xem các thành viên loại Gold thuộc chi nhánh 1 thấy có 20 thành viên, sau đó có đơn hàng của thành viên 23 vừa hoàn thành xong và thành viên đó được hệ thống nâng cấp lên loại Gold. Nhân viên quản lí xem lại thì thấy có thêm một thành viên được loại Gold.

T1	T2
<pre>BEGIN TRAN SELECT * FROM THANHVIEN WHERE LOAITHE='Gold' AND CHINHANHDK=1 waitfor delay'00:00:20'</pre>	<pre>BEGIN TRAN UPDATE THANHVIEN SET LOAITHE='Gold' WHERE MATV=23 COMMIT</pre>

ii. *Chạy lỗi trên SQL*

Chạy đoạn test case gây lỗi như hình 2.3.d.1:

```

27 --Lỗi 3: Phantom
28 BEGIN TRAN
29 SELECT * FROM THANHVIEN
30 WHERE LOAITHE='Gold' AND CHINHANHDK=1
31 waitfor delay '00:00:20'
32
33 SELECT * FROM THANHVIEN
34 WHERE LOAITHE='Gold' AND CHINHANHDK=1
35
36 COMMIT
37
38

```

Results

MATV	TENTV	MATKHAU	CMND	NGAYSINH	DIACHI	CHINHANHDK	EMAIL	DIEMTICHLUY
23	BCD	1	3322233	1999-09-21	Phường 6, Quận 10, TP HCM	1	NULL	0

(1 row(s) affected)

Hình 2.3.d.1

Do cơ chế khóa mặc định là Read Committed nên khi T1 đang giữ khóa S, thì T2 vẫn xin được khóa X. Trong khoảng thời gian 20s, T2 đã chạy truy vấn và thực hiện cập nhật loại thẻ của thành viên 23 từ Silver lên Gold và nhả khóa (commit).

T1 thực hiện tiếp truy vấn đọc thì thấy dữ liệu đã bị thay đổi: Các thành viên đạt hạng Gold từ không ai thành có 1 người là thành viên có mã thành viên 23 (Hình 2.3.d.2).

⇒ Gây ra lỗi Phantom.

```

27 --Lỗi 3: Phantom
28 BEGIN TRAN
29 SELECT * FROM THANHVIEN
30 WHERE LOAITHE='Gold' AND CHINHANHDK=1
31 waitfor delay '00:00:20'
32
33 SELECT * FROM THANHVIEN
34 WHERE LOAITHE='Gold' AND CHINHANHDK=1
35
36 COMMIT
37
38

```

Results

MATV	TENTV	MATKHAU	CMND	NGAYSINH	DIACHI	CHINHANHDK	EMAIL	DIEMTICHLUY
23	BCD	1	3322233	1999-09-21	Phường 6, Quận 10, TP HCM	1	NULL	0

(1 row(s) affected)

```

20
21 --Lỗi 3: Phantom
22
23 BEGIN TRAN
24 UPDATE THANHVIEN
25 SET LOAITHE='Gold'
26 WHERE MATV=23
27
28 COMMIT
29

```

Results

MATV	TENTV	MATKHAU	CMND	NGAYSINH	DIACHI	CHINHANHDK	EMAIL	DIEMTICHLUY
23	BCD	1	3322233	1999-09-21	Phường 6, Quận 10, TP HCM	1	NULL	0

(1 row(s) affected)

Hình 2.3.d.2

iii. Giải pháp: Set cơ chế khóa Serializable

T1	Khóa	T2	Khóa
Set -> Serializable BEGIN TRAN SELECT * FROM THANHVIEN WHERE LOAITHE='Gold' AND CHINHANHDK=1 waitfor delay'00:00:20'	T1: Xin khóa S SQL: Cấp khóa S T1: Giữ khóa S đến hết giao tác T1		
		BEGIN TRAN UPDATE THANHVIEN SET LOAITHE='Gold' WHERE MATV=23 COMMIT	T2: Xin khóa X SQL: Không cấp khóa X do T1 đang giữ khóa S
SELECT * FROM THANHVIEN WHERE LOAITHE='Gold' AND CHINHANHDK=1 COMMIT			

Ta set cơ chế khóa Serializable cho giao tác T1, lúc này T1 sẽ được giữ khóa S cho đến hết giao tác, và T2 không được cấp khóa X cho đến khi giao tác T1 hoàn thành. Các câu truy vấn sẽ thực hiện theo thứ tự: Chạy truy vấn thứ 1 của T1 -> Chạy lệnh waitfor delay '00:00:20' -> Tiếp tục chạy truy vấn thứ 2 của T1 -> Commit T1 -> Chạy truy vấn của T2 -> Commit T2.

Khi đó, T1 không còn gặp lỗi Phantom nữa và đọc được các dữ liệu đúng (Hình 2.3.d.3).

```

Fix1.sql - DELL-PC\DELL (55) ->
34 --Người thực hiện: Trịnh Đức Thành - 1712769
35 --Lỗi 3: Phantom
36 BEGIN TRAN
37 SET TRAN ISOLATION LEVEL SERIALIZABLE
38 SELECT * FROM THANHVIEN
39 WHERE LOAITHE='Gold' AND CHINHANHDK=1
40 waitfor delay'00:00:20'
41
42
43
44
45
46
47
48
49

Fix2.sql - DELL-PC\DELL (52) ->
31 --Người thực hiện: Trịnh Đức Thành - 1712769
32 --Lỗi 3: Phantom
33
34 BEGIN TRAN
35 UPDATE THANHVIEN
36 SET LOAITHE='Gold'
37 WHERE MATV=23
38 COMMIT
39
40
41
42
43
44
45
46

```

Hình 2.3.d.3

e. Lỗi e

❖ **Sinh viên thực hiện:** Dương Khánh Vi – 1712899

i. Kịch bản lỗi

Giả sử nhân viên quản lý khách hàng đang xem lại thông tin của tất cả khách hàng đã đăng ký thành viên trong chi nhánh 2. Trong lúc này, một khách hàng khác vào và đăng ký thành viên trong chi nhánh 2 bằng kênh online. Khi nhân viên xem lại thì thấy số lượng khách hàng đăng ký thành viên trong chi nhánh 2 bị thay đổi.

T1	T2
<pre>BEGIN TRAN SELECT * FROM THANHVIEN WHERE MACN = 2 waitfor delay'00:00:10' SELECT * FROM THANHVIEN WHERE MACN = 2 COMMIT</pre>	<pre>BEGIN TRAN INSERT INTO THANHVIEN VALUES (10,N'ABC','1','3212532','1997-07-07',NULL, 2,'ABC@GMAIL.COM',NULL,N'SILVER') COMMIT</pre>

ii. Chạy lỗi trên SQL

Đoạn test case của trường hợp Phantom: (Giả lập lấy cơ chế Read Committed)

```
SQLQuery3.sql - L_MJ28QOA\Asus (59)* × SQLQuery1.sql - L_MJ28QOA\Asus (55)* × DATA DEMO TRANS_2BQOA\Asus (51))*
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 4: Phantom
BEGIN TRAN
SELECT * FROM THANHVIEN WHERE CHINHANHDK = 2
waitfor delay '00:00:10'
SELECT * FROM THANHVIEN WHERE CHINHANHDK = 2
COMMIT

SQLQuery3.sql - L_MJ28QOA\Asus (60)* × DATA DEMO TRANS_2BQOA\Asus (51))
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 4: Phantom
BEGIN TRAN
INSERT INTO THANHVIEN
VALUES (10,N'ABC','1','3212532','1997-07-07',NULL,
2,'ABC@GMAIL.COM',NULL,N'SILVER')
COMMIT
```

Hình 2.3.e.1

Chạy giao tác 1 đến lệnh waitfor delay '00:00:10', sau đó chạy hết tác 2 và ta có kết quả:

```

SQLQuery3.sql - L...MJ28QOA\Asus (59)* X SQLQuery1.sql - L...MJ28QOA\Asus (55)*
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 4: Phantom
BEGIN TRAN
SELECT * FROM THANHVIEN WHERE CHINHANHDK = 2
waitfor delay '00:00:10'
SELECT * FROM THANHVIEN WHERE CHINHANHDK = 2
COMMIT

SQLQuery3.sql - L...MJ28QOA\Asus (60)* X DATA DEMO TRANS...28QOA\Asus (51)*
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 4: Phantom
BEGIN TRAN
INSERT INTO THANHVIEN
VALUES (10,N'ABC', '1', 3212532, '1997-07-07', NULL,
2,N'ABC@GMAIL.COM',NULL,N'SILVER')
COMMIT

```

MATV	TENTV	MATKHAU	CMND	NGAYSINH	DIACHI	CHINHANHDK	EMAIL	DIEMTICHLUY	LOAITHE	
1	4	BBA	1	321456903	2000-03-17	270 Võ Văn Ngân, Phường 4, Quận Thủ Đức	2	BBA@GMAIL.COM	NULL	SILVER
2	5	BBB	1	321456954	2002-09-20	70 Đường Gia Trí, Phường 8, Quận Bình Thạnh	2	BBB@GMAIL.COM	NULL	SILVER
3	6	BBC	1	321456955	1970-03-17	325 Nguyễn Hữu Cánh, Phường 7, Quận 2	2	BBC@GMAIL.COM	NULL	SILVER
4	10	ABC	1	3212532	1997-07-07	NULL	2	ABC@GMAIL.COM	NULL	SILVER

Hình 2.3.e.2

⇒ Giao tác 1 xem dữ liệu của chi nhánh 2 và nhận được dữ liệu ban đầu (không cập nhật/ thêm). Giao tác 2 đã thêm dữ liệu thành công.

Giao tác 1 tiến hành xem dữ liệu của chi nhánh 2 một lần nữa:

```

SQLQuery3.sql - L...MJ28QOA\Asus (59)* X SQLQuery1.sql - L...MJ28QOA\Asus (55)*
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 4: Phantom
BEGIN TRAN
SELECT * FROM THANHVIEN WHERE CHINHANHDK = 2
waitfor delay '00:00:10'
SELECT * FROM THANHVIEN WHERE CHINHANHDK = 2
COMMIT

SQLQuery3.sql - L...MJ28QOA\Asus (60)* X DATA DEMO TRANS...28QOA\Asus (51)*
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 4: Phantom
BEGIN TRAN
INSERT INTO THANHVIEN
VALUES (10,N'ABC', '1', 3212532, '1997-07-07', NULL,
2,N'ABC@GMAIL.COM',NULL,N'SILVER')
COMMIT

```

MATV	TENTV	MATKHAU	CMND	NGAYSINH	DIACHI	CHINHANHDK	EMAIL	DIEMTICHLUY	LOAITHE	
1	4	BBA	1	321456903	2000-03-17	270 Võ Văn Ngân, Phường 4, Quận Thủ Đức	2	BBA@GMAIL.COM	NULL	SILVER
2	5	BBB	1	321456954	2002-09-20	70 Đường Gia Trí, Phường 8, Quận Bình Thạnh	2	BBB@GMAIL.COM	NULL	SILVER
3	6	BBC	1	321456955	1970-03-17	325 Nguyễn Hữu Cánh, Phường 7, Quận 2	2	BBC@GMAIL.COM	NULL	SILVER
4	10	ABC	1	3212532	1997-07-07	NULL	2	ABC@GMAIL.COM	NULL	SILVER

Query executed successfully.

Hình 2.3.e.3

⇒ Giao tác 1 đọc dữ liệu và thấy kết quả bị thêm 1 dòng.

iii. Giải pháp: Set cơ chế khóa Serializable

T1	Khóa	T2	Khóa
Set -> Serializable			
SELECT * FROM THANHVIEN WHERE MACN = 2 waitfor delay'00:00:10'	T1: xin X(TANHVIEN) SQL: cấp xin X(TANHVIEN) và giữ S cho hết giao tác, ngăn chặn thêm dữ liệu thỏa điều kiện		
		INSERT INTO THANHVIEN VALUES (10,N'ABC','1','3212532','1997-07-07',NULL,2,'ABC@GMAIL.COM',NULL,N'SILVER')	T2: xin X(TANHVIEN) SQL: Rỗng
		COMMIT	
SELECT * FROM THANHVIEN WHERE MACN = 2			
COMMIT			

Chạy giao tác 1 đến lệnh waitfor delay '00:00:15', sau đó chạy hết giao tác 2, lúc này T2 không thêm được dữ liệu do T1 set mức độ cô lập Serializable. Khóa S đã được giữ cho đến hết giao tác, không cho insert dòng thỏa điều kiện thiết lập khóa, nên T2 muốn thêm dữ liệu thì phải đợi T1 commit.

```

SQLQuery7.sql - L...MJ28QOA\Asus (51)* TEST CASE LỖI T1...J28QOA\Asus (59)*
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
--Giải pháp lỗi 4: Phantom
BEGIN TRAN
set tran isolation level SERIALIZABLE
SELECT * FROM THANHVIEN WHERE CHINHANHDK = 2
waitfor delay '00:00:10'
SELECT * FROM THANHVIEN WHERE CHINHANHDK = 2
COMMIT

SQLQuery5.sql - L...MJ28QOA\Asus (55)* SQLQuery1.sql - LAP...s (52) Executing...
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 4: Phantom
BEGIN TRAN
INSERT INTO THANHVIEN VALUES (10,N'ABC','1',3212532,'1997-07-07',NULL,2,'ABC@GMAIL.COM',NULL,N'SILVER')
COMMIT

```

Results (1 row)

MATV	TENTV	MATHKHAU	CMND	NGAYSINH	DIACHI	CHINHANHDK
4	BBA	1	321456903	2000-03-17	270 Võ Văn Ngán, Phường 4, Quận Thủ Đức	2

Results (0 rows)

Hình 2.4.e.4

Đây là kết quả sau khi T1 commit:

```

SQLQuery7.sql - L...MJ28QOA\Asus (51)* TEST CASE LỖI T1...MJ28QOA\Asus (59)*
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
--Giải pháp lỗi 4: Phantom
BEGIN TRAN
set tran isolation level SERIALIZABLE
SELECT * FROM THANHVIEEN WHERE CHINHANHDK = 2
waitfor delay '00:00:10'
SELECT * FROM THANHVIEEN WHERE CHINHANHDK = 2
COMMIT

100 % < Messages
Command(s) completed successfully.

SQLQuery5.sql - L...MJ28QOA\Asus (55)* SQLQuery1.sql - L...MJ28QOA\Asus (52)*
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 4: Phantom
BEGIN TRAN
INSERT INTO THANHVIEEN VALUES (10,N'ABC','1','3212532','1997-07-07',NULL,'ABC@MAIL.COM',NULL,N'SILVER')
COMMIT

SELECT * FROM THANHVIEEN WHERE CHINHANHDK = 2

100 % < Results Messages
MATV TENTV MATKHAU CMND NGAYSINH DIACHI CHINHANHDK
1 4 BBA 1 321456903 2000-03-17 270 Võ Văn Ngân, Phường 4, Quận Thủ Đức 2
2 5 BBB 1 321456954 2002-09-20 70 Đường Gia Trí, Phường 8, Quận Bình Thạnh 2
3 6 BBC 1 321456955 1970-03-17 325 Nguyễn Hữu Cánh, Phường 7, Quận 2 2
4 10 ABC 1 3212532 1997-07-07 NULL 2

< Query exec... LAPTOP-NMJ28QOA (12.0 SP1) | LAPTOP-NMJ28QOA\Asus (51) | HuongViet | 00:00:00 | 0 rows | >
< Query exec... LAPTOP-NMJ28QOA (12.0 SP1) | LAPTOP-NMJ28QOA\Asus (52) | HuongViet | 00:00:00 | 4 rows | >

```

Hình 2.4.e.5

⇒ T2 thêm dữ liệu thành công.

2.4. *Lost Update*

a. *Lỗi a*

❖ **Sinh viên thực hiện:** Nguyễn Thị Ngọc Hân – 1712415

i. *Kịch bản lỗi*

Giả sử khách hàng A (ban đầu đã mua món số 2) đặt thêm hàng món số 2 tại chi nhánh 1 với số lượng là 1, Lúc này có 2 nhân viên quản lý khách hàng truy cập vào tài khoản để cập nhật lại chi tiết đơn hàng. Nhân viên 1 tăng số lượng món lên 1, nhân viên 2 cũng tăng số lượng lên 1. Nhân viên 1 nếu lần sau vào xem sẽ thấy không giống thông tin mình đã sửa.

T1	T2
<pre> BEGIN TRAN SELECT SOLUONG FROM CHITIETDONHANG WHERE MAHD=1 AND MAMON=2 waitfor delay'00:00:05' UPDATE DONHANGCHITIET SET SOLUONG=(SELECT SOLUONG </pre>	<pre> BEGIN TRAN SELECT SOLUONG FROM CHITIETDONHANG WHERE MAHD=1 AND MAMON=2 </pre>

<pre> FROM DOHANGCHITIET WHERE MADH=1 AND MAMON=2)+1 WHERE MADH=1 AND MAMON=2 COMMIT </pre>	<pre> UPDATE DONHANGCHITIET SET SOLUONG=(SELECT SOLUONG FROM DOHANGCHITIET WHERE MADH=1 AND MAMON=2)+1 WHERE MADH=1 AND MAMON=2 COMMIT </pre>
---	--

ii. Chạy lỗi trên SQL

Đoạn test case của trường hợp Lost Update: (Giả lập lấy cơ chế Read Committed)

```

SQLQuery1.sql - D...09AIR6\Admin (55)* SQLQuery2.sql - D...09AIR6\Admin (56)* SQLQuery3.sql - D...09AIR6\Admin (55)* SQLQuery2.sql - D...09AIR6\Admin (56)* SQLQuery3.sql - D...09AIR6\Admin (55)*
1 --Nguyễn Thị Ngọc Hân
2 --MSSV: 1712415
3 --T1
4 BEGIN TRAN
5 SELECT SL FROM CHITIETDONHANG
6 WHERE MADH=1 AND MAMON=2
7 waitfor delay '00:00:15'
8
9
10 UPDATE CHITIETDONHANG
11 SET SL=( SELECT SL FROM CHITIETDONHANG WHERE MADH=1 AND MAMON=2)+1
12 WHERE MADH=1 AND MAMON=2
13
14
15
16
17 COMMIT
18

SQLQuery1.sql - D...09AIR6\Admin (55)* SQLQuery2.sql - D...09AIR6\Admin (56)* SQLQuery3.sql - D...09AIR6\Admin (55)*
1 --Nguyễn Thị Ngọc Hân
2 --MSSV: 1712415
3 --T2
4 BEGIN TRAN
5 SELECT SL FROM CHITIETDONHANG
6 WHERE MADH=1 AND MAMON=2
7
8
9
10 UPDATE CHITIETDONHANG
11 SET SL=( SELECT SL FROM CHITIETDONHANG WHERE MADH=1 AND MAMON=2)+1
12 WHERE MADH=1 AND MAMON=2
13
14
15
16 COMMIT

```

Hình 2.4.a.1

Chạy giao tác 1 đến lệnh waitfor delay '00:00:15' , sau đó chạy hết tác 2 và ta có kết quả:

SL
16

SL
16

Hình 2.4.a.2

⇒ 2 giao tác đều đọc được dữ liệu giống nhau, cũng là dữ liệu ban đầu.

Giao tác 1 cập nhật số lượng và comit:

The screenshot shows a SQL Server Management Studio window with three tabs: SQLQuery1.sql, SQLQuery2.sql, and SQLQuery3.sql. The SQLQuery1.sql tab is active and contains the following script:

```
3 --Nguyễn Thị Ngọc Hân
4 --MSSV: 1712415
5 --T1
6 BEGIN TRAN
7 SELECT SL FROM CHITIETDONHANG
8 WHERE MADH=1 AND MAMON=2
9 waitfor delay '00:00:15'
10
11
12 UPDATE CHITIETDONHANG
13 SET SL=( SELECT SL FROM CHITIETDONHANG WHERE MADH=1 AND MAMON=2)+1
14 WHERE MADH=1 AND MAMON=2
15
16
17 COMMIT
18
19
20
```

Below the script, the message pane shows: (1 row(s) affected).

Hình 2.4.a.3

Select lại ta được kết quả được kết quả:

The screenshot shows a SQL Server Management Studio window with three tabs: SQLQuery1.sql, SQLQuery2.sql, and SQLQuery3.sql. The SQLQuery1.sql tab is active and contains the same script as in Figure 2.4.a.3. Below the script, the Results pane displays the following data:

SL
18

Hình 2.4.a.4

⇒ Giao tác 2 đã cập nhật dữ liệu đè lên giao tác 1, nếu người dùng của giao tác 1 vào xem lại dữ liệu sẽ thấy có sự thay đổi.

iii. Giải pháp: Set cơ chế khoá SERIALIZABLE nhưng tạo thành deadlock

T1	KHOÁ	T2	KHOÁ
SET SERIALIZABLE		SET SERIALIZABLE	
BEGIN TRAN SELECT SOLUONG FROM CHITIETDONHANG WHERE MAHD=1 AND MAMON=2 waitfor delay '00:00:05'	T1 xin khoá S(CHITIETDONHANG) SQL: cấp khoá S(CHITIETDONHANG) cho T1 và giữ đến hết giao tác		
		BEGIN TRAN SELECT SOLUONG FROM CHITIETDONHANG WHERE MAHD=1 AND MAMON=2	T2 xin khoá S(CHITIETDONHANG) SQL: cấp khoá S(CHITIETDONHANG) cho T2 và giữ đến hết giao tác
UPDATE DONHANGCHITIET SET SOLUONG=(SELECT SOLUONG FROM DOHANGCHITIET WHERE MADH=1 AND MAMON=2)+1 WHERE MADH=1 AND MAMON=2 CHỜ T2 NHÃ KHOÁ	T1 xin khoá S(CHITIETDONHANG) SQL: rỗng		
		UPDATE DONHANGCHITIET SET SOLUONG=(SELECT SOLUONG FROM DOHANGCHITIET WHERE MADH=1 AND MAMON=2)+1 WHERE MADH=1 AND MAMON=2 CHỜ T1 NHÃ KHOÁ	T2 xin khoá S(CHITIETDONHANG) SQL: rỗng
COMMIT		COMMIT	

Chạy giao tác 1 đến lệnh waitfor delay '00:00:15', sau đó chạy giao tác 2 đến hết phần select ta có kết quả:

```

T1.sql - DESKTOP-...09AIR6\Admin (56)* X T2.sql - DESKTOP-...09AIR6\Admin (55)* X TEST.sql - DESKTO...
1
2
3 --Nguyễn Thị Ngọc Hân
4 --MSSV: 1712415
5 --T1
6 BEGIN TRAN
7 SET TRAN ISOLATION LEVEL SERIALIZABLE
8 SELECT SL FROM CHITIETDONHANG
9 WHERE MADH=1 AND MAMON=2
10 waitfor delay '00:00:15'
11
12 UPDATE CHITIETDONHANG
13 SET SL=( SELECT SL FROM CHITIETDONHANG WHERE MADH=1 AND MAMON=2)+1
14 WHERE MADH=1 AND MAMON=2
15
16
17 COMMIT
100 % Results Messages
1 SL
1 18

T2.sql - DESKTOP-...09AIR6\Admin (55)* X TEST.sql - DESKTO...
1
2 --Nguyễn Thị Ngọc Hân
3 --MSSV: 1712415a
4 --T2
5 BEGIN TRAN
6 SET TRAN ISOLATION LEVEL SERIALIZABLE
7 SELECT SL FROM CHITIETDONHANG
8 WHERE MADH=1 AND MAMON=2
9
10
11
12 UPDATE CHITIETDONHANG
13 SET SL=( SELECT SL FROM CHITIETDONHANG WHERE MADH=1 AND MAMON=2)+1
14 WHERE MADH=1 AND MAMON=2
15
16
17 COMMIT
100 % Results Messages
1 SI
1 18

```

Hình 2.4.a.5

⇒ T1 và T2 đều đọc được dữ liệu.

Chạy tiếp T2 cho đến hết ta có kết quả sau:

The screenshot shows the SQL Server Management Studio interface with two queries open:

- T1.sql - DESKTOP-...09AIR6\Admin (56)***: Contains the following code:

```
1
2 --Nguyễn Thị Ngọc Hân
3 --MSSV: 1712415a
4 --T2
5 BEGIN TRAN
6 SET TRAN ISOLATION LEVEL SERIALIZABLE
7 SELECT S1 FROM CHITIETDONHANG
8 WHERE MADH=1 AND MAMON=2
9
10
11
12 UPDATE CHITIETDONHANG
13 SET SL=( SELECT SL FROM CHITIETDONHANG WHERE MADH=1 AND MAMON=2)+1
14 WHERE MADH=1 AND MAMON=2
15
16
17 COMMIT
```
- T2.sql - DESKTOP-B0...n (55) Executing...***: Contains the following code:

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
```

The bottom navigation bar shows the **Results** tab selected.

Hình 2.4.a.6

- ⇒ T2 không cập nhật được dữ liệu do T1 đang SET mức cô lập SERIALIZABLE nên khoá S được giữ đến cuối giao tác, T2 muốn xin khoá X thì phải đợi T1 hoàn thành xong

Chạy hết T1 ta có kết quả sau:

```

1
2
3 --Nguyễn Thị Ngọc Hân
4 --MSSV: 1712415
5 --T1
6 BEGIN TRAN
7 SET TRAN ISOLATION LEVEL SERIALIZABLE
8 SELECT SL FROM CHITIETDONHANG
9 WHERE MADH=1 AND MAMON=2
10 waitfor delay '00:00:15'
11
12
13 UPDATE CHITIETDONHANG
14 SET SL=( SELECT SL FROM CHITIETDONHANG WHERE MADH=1 AND MAMON=2)+1
15 WHERE MADH=1 AND MAMON=2
16
17
18 COMMIT

```

Messages

Msg 1205, Level 13, State 51, Line 13
Transaction (Process ID 56) was deadlocked on lock resources with another process and has been

Hình 2.4.a.7

- ⇒ Xảy ra deadlock do T1 không cập nhật được dữ liệu do T2 đang SET mức cô lập SERIALIZABLE nên khoá S được giữ đến cuối giao tác, T1 muốn xin khoá X thì phải đợi T2 hoàn thành xong
 - ⇒ T1 và T2 chờ nhau nên xảy ra deadlock
- b. *Lỗi b*
- ❖ **Sinh viên thực hiện:** Lai Gia Phú – 1712662
 - i. *Kịch bản lỗi*

Giả sử quản lý chi nhánh 1 vào xem thực đơn chi nhánh 1 sau đó cập nhật số phần còn lại của món 1 là 45. Cùng lúc đó quản lý công ty vào xem thực đơn chi nhánh 1, sau đó cập nhật số phần còn lại của món 1 là 40.Nếu sau đó quản lý chi nhánh vào xem lại thực đơn thì sẽ không hiện thông tin như mình vừa cập nhật.

T1	T2
BEGIN TRAN EXEC sp_XemThucDon 1,'2020-01-01'	

<pre>Waitfor delay '00:00:10'</pre> <pre>EXEC sp_CapNhatSoPhanAn 1,1,'2020-01-01',50,45</pre> <pre>COMMIT TRAN</pre>	<pre>BEGIN TRAN</pre> <pre>EXEC sp_XemThucDon 1,'2020-01-01'</pre> <pre>EXEC sp_CapNhatSoPhanAn 1,1,'2020-01-01',50,40</pre> <pre>COMMIT TRAN</pre>
--	--

ii. Chạy lỗi trên SQL

Đoạn test case của trường hợp Lost Update: (Giả lập lấy cơ chế Read Committed).

The screenshot shows two SQL queries running in separate windows:

- Testcase03.sql:**

```
--Lost Update - Lai Gia Phú - 1712662
--T1
BEGIN TRAN
EXEC sp_XemThucDon 1, '2020-01-01'
Waitfor delay '00:00:10'
EXEC sp_CapNhatSoPhanAn 1,1,'2020-01-01',50,45
COMMIT TRAN
```
- SQLQuery16.sql:**

```
--T2
BEGIN TRAN
EXEC sp_XemThucDon 1, '2020-01-01'
EXEC sp_CapNhatSoPhanAn 1,1,'2020-01-01',50,40
COMMIT TRAN
```

Both queries return the same result set:

	MAMON	TENMON	TENLOAI	SL_CONLAI	SOLUONG
1	1	Phở	MÓN NƯ...	50	50

Hình 2.4.b.1

⇒ 2 giao tác đều đọc được dữ liệu giống nhau, cũng là dữ liệu ban đầu.

Dữ liệu thật sự sau cập nhật

```

--Lost Update - Lai Gia Phú - 1712662
--T1
BEGIN TRAN
EXEC sp_XemThucDon 1, '2020-01-01'
Waitfor delay '00:00:10'
EXEC sp_CapNhatSoPhanAn 1,1, '2020-01-01', 50, 45
COMMIT TRAN

```

MAMON	TENMON	TENLOAI	SL_CONLAI	SOLUONG
1	PHở	MÓN NƯỚNG	45	50

Hình 2.4.b.2

- ⇒ Giao tác 1 đã cập nhật dữ liệu đè lên giao tác 2, nếu người dùng của giao tác 2 vào xem lại dữ liệu sẽ không giống như dữ liệu họ cập nhật.
- iii. *Giải pháp:* Thiết lập mức độ lặp Repeatable Read ở T1 và T2. Xảy ra deadlock và SQL sẽ ngừng T2 và bắt thực hiện lại sau.

```

--T1
BEGIN TRAN
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ
EXEC sp_XemThucDon 1, '2020-01-01'
Waitfor delay '00:00:10'
EXEC sp_CapNhatSoPhanAn 1,1, '2020-01-01', 50, 45
COMMIT TRAN

```

MAMON	TENMON	TENLOAI	SL_CONLAI	SOLUONG
1	PHở	MÓN NƯỚNG	50	50


```

--T2
BEGIN TRAN
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ
EXEC sp_XemThucDon 1, '2020-01-01'
EXEC sp_CapNhatSoPhanAn 1,1, '2020-01-01', 50, 40
COMMIT TRAN

```

MAMON	TENMON	TENLOAI	SL_CONLAI	SOLUONG
1	PHở	MÓN NƯỚNG	50	50

(1 row affected)

Msg 1205, Level 19, State 51, Procedure sp_CapNhatSoPhanAn, Line 20 (Batch Start Line 0)
Transaction (Process ID 53) was deadlocked on lock resources with another process and has been chosen as the deadlock victim.

Completion time: 2020-01-02T11:12:47.1650667#07:00

Hình 2.4.b.3

Đồ thi ưu tiên: T1 → T2

c. Lỗi c

❖ **Sinh viên thực hiện:** Nguyễn Đoàn Tấn Phúc – 1712671

i. Kịch bản lỗi

T1 đăng nhập và thay đổi giá trị thẻ, sau đó T2 đăng nhập và cũng thay đổi giá trị thẻ. Qua đó update của T1 bị mất.

T1	T2
BEGIN TRAN EXEC SP_DangNhap 1,1 Waitfor delay '00:00:10' EXEC sp_UPTHETV3 1,1,2 COMMIT	BEGIN TRAN EXEC SP_DangNhap 1,1 Waitfor delay '00:00:10' EXEC sp_UPTHETV3 1,1,3 COMMIT

ii. Chạy lỗi trên SQL

Đoạn test case của trường hợp Lost Update: (Giả lập lấy cơ chế Read Committed).

```

-- T1
BEGIN TRAN
select * from THANHVIEN where MATV = 1
Waitfor delay '00:00:10'
EXEC sp_UPTHETV3 1,1,2
COMMIT TRAN

-- T2
BEGIN TRAN
select * from THANHVIEN where MATV = 1
Waitfor delay '00:00:10'
EXEC sp_UPTHETV3 1,1,3
COMMIT TRAN

-- Sửa : mức cờ lặp Serializable ở T1 nhưng gặp Deadlock và thực hiện lại T2
-- T1
BEGIN TRAN
set tran isolation level Serializable
select * from THANHVIEN where MATV = 1
Waitfor delay '00:00:10'
EXEC sp_UPTHETV3 1,1,2
COMMIT TRAN

select * from THANHVIEN where MATV = 1

```

Hình 2.4.c.1

Chạy giao tác 1 , sau đó chạy hết tác 2 và ta có kết quả:

MATV	TEN...	MATKH...	CMND	NGAYSINH	DIACHI	CHINHANH
1	AAA	1	321456984	2000-03-17	112 Nguyễn Văn Cừ, Phường 6, Quận 5	1

MATV	TEN...	MATKH...	CMND	NGAYSINH	DIACHI	CHINHANHDK	EN
1	AAA	1	321456984	2000-03-17	112 Nguyễn Văn Cừ, Phường 6, Quận 5	1	A

Hình 2.4.c.2

⇒ 2 giao tác đều đọc được dữ liệu giống nhau, cũng là dữ liệu ban đầu.

Giao tác 1 cập nhật mật khẩu và comit:

⇒ Dữ liệu đã được cập nhật thành công

Giao tác 2 cập nhật địa chỉ và commit:

The screenshot shows two separate sessions in SQL Server Management Studio. Both sessions run the same code:

```
--T1
BEGIN TRAN
select * from THANHVIEN where MATV = 1
Waitfor delay '00:00:10'
EXEC sp_UPTHETV3 1,1,2
COMMIT TRAN
```

The results show different outcomes for each session:

MATV	TEN...	MATKH...	CMND	NGAYSINH	DIACHI	CHINHANH
1	AAA	1	321456984	2000-03-17	112 Nguyễn Văn Cừ, Phường 6, Quận 5	1

Session 1 (left) shows MATV = 1. Session 2 (right) shows MATV = 3.

Hình 2.4.c.3

- ⇒ Giao tác 2 đã cập nhật dữ liệu đè lên giao tác 1, nếu người dùng của giao tác 1 vào xem lại dữ liệu sẽ thấy có sự thay đổi. Ở đây là pass = 3 không là pass = 2 như T1 mong muốn.

iii. Giải pháp: Set cơ chế khóa Serializable.

The screenshot shows the same setup as above, but with a set tran isolation level Serializable added to the code:

```
--T1
set tran isolation level Serializable
select * from THANHVIEN where MATV = 1
Waitfor delay '00:00:10'
EXEC sp_UPTHETV3 1,1,2
COMMIT TRAN
```

The results are now consistent across both sessions:

MATV	TEN...	MATKH...	CMND	NGAYSINH	DIACHI	CHINHANH
1	AAA	1	321456984	2000-03-17	112 Nguyễn Văn Cừ, Phường 6, Quận 5	1

Both Session 1 and Session 2 now show MATV = 1.

Hình 2.4.c.4

Pass của thành viên có mã là 1 đã thành 2 không còn là 3 như trước khi set khóa Serializable.

d. Lỗi d

❖ **Sinh viên thực hiện:** Trịnh Đức Thanh – 1712769

i. Kịch bản lỗi

Một nhân viên quản lý khánh hàng 1 của chi nhánh 1 tạo đơn hàng và update số lượng món 1 trong bảng thực đơn từ 30 thành 19, cùng lúc đó một nhân viên quản lý khánh hàng 2 của chi nhánh 1 tạo đơn hàng khác và update số lượng món 1 trong bảng thực đơn từ 30 thành 17. Khi nhân viên quản lý khánh hàng 2 xem lại thực đơn thì thấy số lượng món ăn 1 sai.

T1	T2
<pre>BEGIN TRAN SELECT SL_CONLAI FROM THUCDON WHERE MAMON = 1 AND MACN = 1 waitfor delay'00:00:20'</pre>	<pre>BEGIN TRAN SELECT SL_CONLAI FROM THUCDON WHERE MAMON = 1 AND MACN = 1</pre>

ii. Chạy lỗi trên SQL

Chạy đoạn test case gây lỗi như hình 2.4.d.1:

```

Test1.sql - DELL-PC\DELL (56)* × Fix1.sql - DELL-PC\DELL (55)
37
38 --Người thực hiện: Trịnh Đức Thành - 1712769
39
40 --Lỗi 4: Lost Update
41 BEGIN TRAN
42 SELECT SL_CONLAI
43 FROM THUCDON
44 WHERE MAMON = 1 AND MACN = 1
45 waitfor delay'00:00:20'
46
47 UPDATE THUCDON
48 SET SL_CONLAI = 19
49 WHERE MAMON = 1 AND MACN = 1
50 COMMIT
51
52

Test2.sql - DELL-PC\DELL (57)* × Fix2.sql - DELL-PC\DELL (52)*
28
29 --Người thực hiện: Trịnh Đức Thành - 1712769
30
31 --Lỗi 4: Lost Update
32 BEGIN TRAN
33 SELECT SL_CONLAI
34 FROM THUCDON
35 WHERE MAMON = 1 AND MACN = 1
36
37 UPDATE THUCDON
38 SET SL_CONLAI = 17
39 WHERE MAMON = 1 AND MACN = 1
40 COMMIT
41
42 SELECT SL_CONLAI FROM THUCDON WHERE MAMON = 1 AND MACN = 1
43

```

Hình 2.4.d.1

Do cơ chế khóa mặc định là Read Committed nên khi T1 đang giữ khóa S, thì T2 vẫn xin được khóa S. Trong 20s, T2 xin khóa S và thực hiện truy vấn đọc số lượng còn lại của món ăn 1. Sau đó, T2 tiếp tục xin khóa X để thực hiện truy vấn cập nhật số lượng còn lại của món ăn 1 từ 30 thành 17. Sau khi thực hiện xong, T2 nhả khóa (commit).

T1 lúc này tiếp tục thực hiện truy vấn sau lệnh `waitfor delay '00:00:20'` và cập nhật lại số lượng còn lại của món ăn 1 từ 30 thành 19, điều này làm cho dữ liệu T2 cập nhật trước đó bị T1 ghi chồng lên dẫn đến dữ liệu sai.

⇒ Gây ra lỗi Lost Update.

Khi T2 xem lại số lượng còn lại của món ăn 1 thì thấy dữ liệu bị sai (Hình 2.4.d.2).

SL_CONLAI
19

Hình 2.4.d.2

iii. Giải pháp: Set cơ chế khóa Repeatable Read (hoặc Serializable) nhưng sẽ tạo thành Deadlock

T1	Khóa	T2	Khóa
Set -> Repeatable Read BEGIN TRAN SELECT SOLUONGCONLAI FROM THUCDON WHERE MAMON = 1 AND MACN = 1 waitfor delay'00:00:20'	T1: Xin khóa S SQL: Cấp khóa S T1: Giữ khóa S đến hết giao tác T1	Set -> Repeatable Read	
		BEGIN TRAN SELECT SOLUONGCONLAI FROM THUCDON WHERE MAMON = 1 AND MACN = 1	T2: Xin khóa S SQL: Cấp khóa S T2: Giữ khóa S đến hết giao tác T2
UPDATE THUCDON SET SOLUONGCONLAI =19 WHERE MAMON = 1 AND MACN = 1	T1: Xin khóa X SQL: Không cấp khóa X do phải chờ T2 nhả khóa S	Chờ	
COMMIT			
	Chờ	UPDATE THUCDON SET SOLUONGCONLAI =17 WHERE MAMON = 1 AND MACN = 1	T2: Xin khóa X SQL: Không cấp khóa X do chờ T1 nhả khóa X
		COMMIT	

Ta set cơ chế khóa Repeatable Read cho cả giao tác T1 và T2, lúc này T1 sẽ được giữ khóa S cho đến hết giao tác, và T2 không được cấp khóa X cho đến khi giao tác T1 hoàn thành.

Khi đó, T1 không còn gặp lỗi Lost Update nữa và T2 sẽ đọc được các dữ liệu đúng.

Cả T1 và T2 đều đọc được dữ liệu ban đầu là số lượng còn lại của món ăn 1 là 30 (Hình 2.4.d.3).

```

Test1.sql - DELL-P...DELL (56)* Fix1.sql - DELL-PC...(DELL-PC\DELL (55))*
52 --Người thực hiện: Trịnh Đức Thành - 1712769
53 BEGIN TRAN
54 SET TRAN ISOLATION LEVEL REPEATABLE READ
55 SELECT SL_CONLAI
56 FROM THUCDON
57 WHERE MANON = 1 AND MACN = 1
58 waitfor delay'00:00:20'
59 UPDATE THUCDON
60 SET SL_CONLAI = 19
61 WHERE MANON = 1 AND MACN = 1
62 COMMIT
63
64
65
66
67

Test2.sql - DELL-P...DELL (57)* Fix2.sql - DELL-PC...(DELL-PC\DELL (52))*
46 --Người thực hiện: Trịnh Đức Thành - 1712769
47 BEGIN TRAN
48 SET TRAN ISOLATION LEVEL REPEATABLE READ
49 SELECT SL_CONLAI
50 FROM THUCDON
51 WHERE MANON = 1 AND MACN = 1
52 UPDATE THUCDON
53 SET SL_CONLAI = 17
54 WHERE MANON = 1 AND MACN = 1
55 COMMIT
56
57
58
59
60
61

```

Results (Test1.sql): SL_CONLAI
1 30

Results (Test2.sql): SL_CONLAI
1 30

Hình 2.4.d.3

Sau đó cả hai bị deadlock và giao tác T2 bị buộc rollback (Hình 2.3.d.4), và giao tác T1 cập nhật thành công số lượng còn lại của món ăn 1 từ 30 thành 19.

```

Test1.sql - DELL-P...DELL (56)* Fix1.sql - DELL-PC...(DELL-PC\DELL (55))*
52 --Người thực hiện: Trịnh Đức Thành - 1712769
53 BEGIN TRAN
54 SET TRAN ISOLATION LEVEL REPEATABLE READ
55 SELECT SL_CONLAI
56 FROM THUCDON
57 WHERE MANON = 1 AND MACN = 1
58 waitfor delay'00:00:20'
59 UPDATE THUCDON
60 SET SL_CONLAI = 19
61 WHERE MANON = 1 AND MACN = 1
62 COMMIT
63
64
65
66
67

Test2.sql - DELL-P...DELL (57)* Fix2.sql - DELL-PC...(DELL-PC\DELL (52))*
46 --Người thực hiện: Trịnh Đức Thành - 1712769
47 BEGIN TRAN
48 SET TRAN ISOLATION LEVEL REPEATABLE READ
49 SELECT SL_CONLAI
50 FROM THUCDON
51 WHERE MANON = 1 AND MACN = 1
52 UPDATE THUCDON
53 SET SL_CONLAI = 17
54 WHERE MANON = 1 AND MACN = 1
55 COMMIT
56
57
58
59
60
61

```

Results (Test1.sql): (1 row(s) affected)
(1 row(s) affected)

Messages (Test2.sql): (1 row(s) affected)
Msg 1205, Level 13, State 51, Line 57
Transaction (Process ID 52) was deadlocked on lock resources with another process and has been rolled back.

Hình 2.4.d.4

Lúc này T2 thực hiện lại giao tác thì thấy số lượng còn lại của món ăn 1 đã thành 19 (Hình 2.4.d.5).

```

52   --Người thực hiện: Trịnh Đức Thành - 1712769
53
54   --Lỗi 4: Lost Update
55   BEGIN TRAN
56   SET TRAN ISOLATION LEVEL REPEATABLE READ
57   SELECT SL_CONLAI
58   FROM THUCDON
59   WHERE MAMON = 1 AND MACN = 1
60   waitfor delay'00:00:20'
61
62   UPDATE THUCDON
63   SET SL_CONLAI =19
64   WHERE MAMON = 1 AND MACN = 1
65
66   COMMIT
67

```

(1 row(s) affected)
(1 row(s) affected)

```

47   --Người thực hiện: Trịnh Đức Thành - 1712769
48
49   --Lỗi 4: Lost Update
50   BEGIN TRAN
51   SET TRAN ISOLATION LEVEL REPEATABLE READ
52   SELECT SL_CONLAI
53   FROM THUCDON
54   WHERE MAMON = 1 AND MACN = 1
55
56   UPDATE THUCDON
57   SET SL_CONLAI =17
58   WHERE MAMON = 1 AND MACN = 1
59
60   COMMIT
61
62

```

SL_CONLAI
1 19

Hình 2.4.d.5

Sau khi T2 thực hiện xong ta thử truy vấn xem số lượng món ăn 1 thì thấy đã thành 17 (Hình 2.4.d.6). Như vậy, đã không còn lỗi Lost Update nữa và các dữ liệu được đọc đều đúng.

```

52   --Người thực hiện: Trịnh Đức Thành - 1712769
53
54   --Lỗi 4: Lost Update
55   BEGIN TRAN
56   SET TRAN ISOLATION LEVEL REPEATABLE READ
57   SELECT SL_CONLAI
58   FROM THUCDON
59   WHERE MAMON = 1 AND MACN = 1
60   waitfor delay'00:00:20'
61
62   UPDATE THUCDON
63   SET SL_CONLAI =19
64   WHERE MAMON = 1 AND MACN = 1
65
66   COMMIT
67

```

(1 row(s) affected)
(1 row(s) affected)

```

47   --Người thực hiện: Trịnh Đức Thành - 1712769
48
49   --Lỗi 4: Lost Update
50   BEGIN TRAN
51   SET TRAN ISOLATION LEVEL REPEATABLE READ
52   SELECT SL_CONLAI
53   FROM THUCDON
54   WHERE MAMON = 1 AND MACN = 1
55
56   UPDATE THUCDON
57   SET SL_CONLAI =17
58   WHERE MAMON = 1 AND MACN = 1
59
60   COMMIT
61
62   SELECT SL_CONLAI FROM THUCDON WHERE MAMON = 1 AND MACN = 1

```

SL_CONLAI
1 17

Hình 2.4.d.5

e. Lỗi e

❖ **Sinh viên thực hiện:** Dương Khánh Vi – 1712899

i. Kịch bản lỗi

Giả sử có 2 người đồng thời truy cập vào tài khoản thành viên 1. Người số 1 vào cập nhật lại địa chỉ là 'số 7, Nguyễn Hữu Cánh, phường 3, quận 2, TPHCM', người số 2 vào cũng cập nhật lại địa chỉ '122, Nguyễn Văn Cừ, phường 10,

quận 5, TPHCM', người thứ 1 nếu lần sau vào xem lại thông tin thì sẽ không giống với thông tin ban đầu mình đã sửa nữa.

T1	T2
<pre>BEGIN TRAN SELECT DIACHI FROM THANHVIEN WHERE MATV = 1 waitfor delay'00:00:10' UPDATE THANHVIEN SET DIACHI = N 'Khu đô thị Sala, phường 3, quận 2, TPHCM' COMMIT</pre>	<pre>BEGIN TRAN SELECT DIACHI FROM THANHVIEN WHERE MATV = 1 UPDATE THANHVIEN SET DIACHI = N' 122, Nguyễn Văn Cừ, phường 10, quận 5, TPHCM' COMMIT</pre>

ii. Chạy lỗi trên SQL

Đoạn test case của trường hợp Lost Update: (Giả lập lấy cơ chế Read Committed).

```

TEST CASE LỖI T1....J28QOA\Asus (58)* X Giải pháp lỗi T2.s..Mj28QOA\Asus (54)*
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 3: Lost update
BEGIN TRAN
SELECT DIACHI FROM THANHVIEN WHERE MATV = 1
waitfor delay '00:00:10'
UPDATE THANHVIEN
SET DIACHI = N'Khu đô thị Sala, phường 3, quận 2, TPHCM' WHERE MATV = 1
COMMIT
GO

TEST CASE LỖI T2....J28QOA\Asus (59)* X Giải pháp lỗi T1.s..Mj28QOA\Asus (55)
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 3: Lost Update
BEGIN TRAN
SELECT DIACHI FROM THANHVIEN WHERE MATV = 1
UPDATE THANHVIEN
SET DIACHI = N'122, Nguyễn Văn Cử, phường 10, quận 5, TPHCM' WHERE MATV = 1
COMMIT
GO

```

Query executed... | LAPTOP-NMJ28QOA (12.0 SP1) | LAPTOP-NMJ28QOA\Asus (58) | HuongViet | 00:00:00 | 0 rows

Query executed... | LAPTOP-NMJ28QOA (12.0 SP1) | LAPTOP-NMJ28QOA\Asus (59) | HuongViet | 00:00:00 | 1 rows

Hình 2.4.e.1

Chạy giao tác 1 đến lệnh `waitfor delay '00:00:10'`, sau đó chạy lệnh `Select` của giao tác 2 và ta có kết quả:

```

TEST CASE LỖI T1....J28QOA\Asus (58)* X Giải pháp lỗi T2.s..Mj28QOA\Asus (54)*
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 3: Lost update
BEGIN TRAN
SELECT DIACHI FROM THANHVIEN WHERE MATV = 1
waitfor delay '00:00:10'
UPDATE THANHVIEN
SET DIACHI = N'Khu đô thị Sala, phường 3, quận 2, TPHCM' WHERE MATV = 1
COMMIT
GO

TEST CASE LỖI T2....J28QOA\Asus (59)* X Giải pháp lỗi T1.s..Mj28QOA\Asus (55)
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 3: Lost Update
BEGIN TRAN
SELECT DIACHI FROM THANHVIEN WHERE MATV = 1
UPDATE THANHVIEN
SET DIACHI = N'122, Nguyễn Văn Cử, phường 10, quận 5, TPHCM' WHERE MATV = 1
COMMIT
GO

```

Results Messages

DIACHI
1 390 Nguyễn Văn Cử, phường 10, quận 5, TPHCM

Results Messages

DIACHI
1 390 Nguyễn Văn Cử, phường 10, quận 5, TPHCM

Query executed... | LAPTOP-NMJ28QOA (12.0 SP1) | LAPTOP-NMJ28QOA\Asus (58) | HuongViet | 00:00:10 | 1 rows

Query... | LAPTOP-NMJ28QOA (12.0 SP1) | LAPTOP-NMJ28QOA\Asus (59) | HuongViet | 00:00:00 | 1 rows

Hình 2.4.e.2

⇒ 2 giao tác đều đọc được dữ liệu giống nhau, cũng là dữ liệu ban đầu.

Giao tác 1 cập nhật địa chỉ và commit:

```

USE HuongViet
-- Lỗi 3: Lost update
BEGIN TRAN
SELECT DIACHI FROM THANHVIEN WHERE MATV = 1
waitfor delay '00:00:10'
UPDATE THANHVIEN
SET DIACHI = N'Khu đô thị Sala, phường 3, quận 2, TPHCM' WHERE MATV = 1
COMMIT

SELECT DIACHI FROM THANHVIEN WHERE MATV = 1

```

Results

DIACHI
Khu đô thị Sala, phường 3, quận 2, TPHCM

Hình 2.4.e.3

⇒ Dữ liệu đã được cập nhật thành công

Giao tác 2 cập nhật địa chỉ và commit:

```

/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 3: Lost update
BEGIN TRAN
SELECT DIACHI FROM THANHVIEN WHERE MATV = 1
UPDATE THANHVIEN
SET DIACHI = N'122, Nguyễn Văn Cừ, phường 10, quận 5, TPHCM' WHERE MATV = 1
COMMIT

SELECT DIACHI FROM THANHVIEN WHERE MATV = 1

```

Results

DIACHI
122, Nguyễn Văn Cừ, phường 10, quận 5, TPHCM

Hình 2.4.e.4

⇒ Giao tác 2 đã cập nhật dữ liệu đè lên giao tác 1, nếu người dùng của giao tác 1 vào xem lại dữ liệu sẽ thấy có sự thay đổi.

iii. Giải pháp: Set cơ chế khóa Serializable nhưng sẽ tạo thành Deadlock.

T1	Khóa	T2	Khóa
Set -> Serializable		Set -> Serializable	
SELECT DIACHI FROM THANHVIEN WHERE MATV = 1	<u>T1</u> : Xin khóa S <u>SQL</u> : Cấp khóa S <u>T1</u> : giữ khóa S đến hết giao tác, ngăn chèn dữ liệu vào tập đang khóa		
		SELECT DIACHI FROM THANHVIEN WHERE MATV = 1	
UPDATE THANHVIEN SET DIACHI = N 'Khu đô thị Sala, phường 3, quận 2, TPHCM'	<u>T1</u> : Xin khóa X <u>SQL</u> : Không cấp khóa X cho T1 do T2 đang giữ khóa S		Chờ
			Chờ
		UPDATE THANHVIEN SET DIACHI = N' 122, Nguyễn Văn Cừ, phường 10, quận 5, TPHCM' Chờ T1 nhả khóa	<u>T2</u> : Xin khóa X <u>SQL</u> : Không cấp khóa X cho T2 do T1 đang giữ khóa X
COMMIT		COMMIT	

Chạy giao tác 1 đến lệnh waitfor delay '00:00:10', sau đó chạy lệnh Select của giao tác 2, lúc này T2 xin khóa S và đọc được dữ liệu chung với T1. Khóa S của T1 và T2 đã được giữ cho đến hết giao tác. Tiếp theo chạy lệnh update của T1, T1 xin khóa X nhưng bị từ chối vì phải đợi T2 thực hiện hết giao tác và nhả khóa S. Lệnh update của T2 được chạy, T2 xin khóa X cũng bị từ chối vì phải đợi T1 nhả khóa X, trong khi đó T1 cũng đang đợi T2 nhả khóa S.

Trường hợp này Deadlock đã xảy ra.

The screenshot shows two separate SSMS sessions. The left session, titled 'TEST CASE Lỗi T1..._L28QOA\Asus (58)*', contains the following script:

```
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 3: Lost update
BEGIN TRAN
set tran isolation level SERIALIZABLE
SELECT DIACHI FROM THANHVIEN WHERE MATV = 1
waitfor delay '00:00:10'
UPDATE THANHVIEN
SET DIACHI = N'Khu đô thị Sala, phường 3, quận 2, TPHCM' WHERE MATV = 1
COMMIT
```

The right session, titled 'TEST CASE Lỗi T2..._L28QOA\Asus (59)*', contains the following script:

```
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
BEGIN TRAN
set tran isolation level SERIALIZABLE
SELECT DIACHI FROM THANHVIEN WHERE MATV = 1
UPDATE THANHVIEN
SET DIACHI = N'122, Nguyễn Văn Cừ, phường 10, quận 5, TPHCM' WHERE MATV = 1
COMMIT
```

Both sessions show a 'Messages' pane at the bottom with the following error message:

Msg 1205, Level 13, State 51, Line 6
Transaction (Process ID 59) was deadlocked on lock resources with another process and has been killed.

Hình 24 e 5

2.5. *Deadlock*

a / $\tilde{O}(n)$

❖ **Sinh viên thực hiện:** Nguyễn Thị Ngọc Hân – 1712415

i Kích bản lõi

Giả sử khách hàng A (ban đầu đã mua món số 2) đặt thêm hàng món số 2 tại chi nhánh 1 với số lượng là 1, Lúc này có 2 nhân viên quản lý khách hàng truy cập vào tài khoản để cập nhật lại chi tiết đơn hàng. Nhân viên 1 tăng số lượng món lên 1, nhân viên 2 cũng tăng số lượng lên 1. Nhân viên 1 nếu lần sau vào xem sẽ thấy không giống thông tin mình đã sửa. Để xảy ra deadlock ta xét mức độ SERIALIZABLE.

T1	T2
<pre>BEGIN TRAN SET TRAN ISOLATION LEVEL SERIALIZABLE SELECT SOLUONG FROM CHITIETDONHANG WHERE MAHD=1 AND MAMON=2 waitfor delay'00:00:05' UPDATE DONHANGCHITIET SET SOLUONG=(SELECT SOLUONG FROM DOHANGCHITIET WHERE MADH=1 AND MAMON=2)+1 WHERE MADH=1 AND MAMON=2 COMMIT</pre>	<pre>BEGIN TRAN SET TRAN ISOLATION LEVEL SERIALIZABLE SELECT SOLUONG FROM CHITIETDONHANG WHERE MAHD=1 AND MAMON=2 UPDATE DONHANGCHITIET SET SOLUONG=(SELECT SOLUONG FROM DOHANGCHITIET WHERE MADH=1 AND MAMON=2)+1 WHERE MADH=1 AND MAMON=2 COMMIT</pre>

ii. Chạy lỗi trên SQL

Thực hiện tương tự câu lỗi lost update ta được kết quả sau:

The screenshot shows three separate SQL Server Management Studio windows. The first window (T1.sql) contains a transaction T1 that selects a row from CHITIETDONHANG where MADH=1 and MAMON=2, then waits for 15 seconds. The second window (T2.sql) contains a transaction T2 that updates the same row in CHITIETDONHANG where MADH=1 and MAMON=2. Both transactions are set to serializable isolation level. The third window (TEST.sql) shows the results of the update, indicating 1 row affected. A message in the bottom-left window states: "Msg 1205, Level 13, State 51, Line 13 Transaction (Process ID 56) was deadlocked on lock resources with another process and has been chosen as the deadlock victim. Rerun the transaction."

Hình 2.5.a.1

⇒ Xảy ra lỗi deadlock.

iii. Giải pháp

Để giải quyết deadlock ta huỷ 1 giao tác chạy 1 giao t, sau đó thực hiện lại giao tác bị huỷ. Ở đây thì SQL đã tự động huỷ T1 nên T2 update dữ liệu thành công, T1 bị huỷ.

This screenshot shows a single SQL Server Management Studio window with three tabs: T1.sql, T2.sql, and TEST.sql. The T1.sql tab contains a transaction T1 with the same code as in the previous screenshot. The T2.sql tab contains a transaction T2 that updates the same row in CHITIETDONHANG. The TEST.sql tab shows the update was successful with 1 row affected. A message in the bottom-left window states: "Msg 1205, Level 13, State 51, Line 13 Transaction (Process ID 56) was deadlocked on lock resources with another process and has been chosen as the deadlock victim. Rerun the transaction." This indicates that the deadlock was resolved by rolling back transaction T1.

Hình 2.5.a.2

b. Lỗi b

❖ **Sinh viên thực hiện:** Lai Gia Phú – 1712662

i. Kịch bản lỗi

Giả sử có 2 người quản lý công ty. Người thứ nhất cập nhật số phần ăn của món 1 ở chi nhánh 1 sau đó tiến hành xem thực đơn chi nhánh 2 cần chỉnh sửa gì không. Cùng lúc đó người thứ 2 cập nhật số phần ăn của món 1 ở chi nhánh

2 và sau đó tiến hành xem thực đơn chi nhánh 1. Lúc này giao tác của người thứ 2 bị rollback và thực hiện lại theo cơ chế xử lý Deadlock của SQL.

T1	T2
<pre>BEGIN TRAN EXEC sp_CapNhatSoPhanAn 1,1,'2019-01-01',50,40 Waitfor delay '00:00:05' EXEC sp_XemThucDon 2,'2019-01-01' COMMIT TRAN</pre>	<pre>BEGIN TRAN EXEC sp_CapNhatSoPhanAn 2,1,'2019-01-01',50,30 Waitfor delay '00:00:10' EXEC sp_XemThucDon 1,'2019-01-01' COMMIT TRAN</pre>

ii. Chạy lỗi trên SQL

Đoạn test case của trường hợp Deadlock: (Giả lập lấy cơ chế Read Committed)

```
TestCase05.sql - D...\MEST3H\dell (52)* - Deadlock - Lai Gia Phú - 1712662
--T1
BEGIN TRAN
EXEC sp_CapNhatSoPhanAn 1,1,'2019-01-01',50,40
Waitfor delay '00:00:05'
EXEC sp_XemThucDon 2,'2019-01-01'
COMMIT TRAN

SQLQuery15.sql - D...\MEST3H\dell (53)* - T2
--T2
BEGIN TRAN
EXEC sp_CapNhatSoPhanAn 2,1,'2019-01-01',50,30
Waitfor delay '00:00:10'
EXEC sp_XemThucDon 1,'2019-01-01'
COMMIT TRAN
```

Hình 2.5.b.1

Đồ thi ưu tiên: T1 -> T2.

iii. Giải pháp

SQL chọn T2 là victim Deadlock và bắt thực hiện lại sau.

```
Sửa thành công
Msg 1205, Level 13, State 51, Procedure sp_XemThucDon, Line 7 [Batch Start Line 1]
Transaction (Process ID 53) was deadlocked on lock resources with another process and has been chosen as the deadlock victim. Rerun the transaction.

Completion time: 2020-01-02T10:46:09.5055619+07:00
```

Hình 2.5.b.2

c. Lỗi c

❖ **Sinh viên thực hiện:** Nguyễn Đoàn Tấn Phúc – 1712671

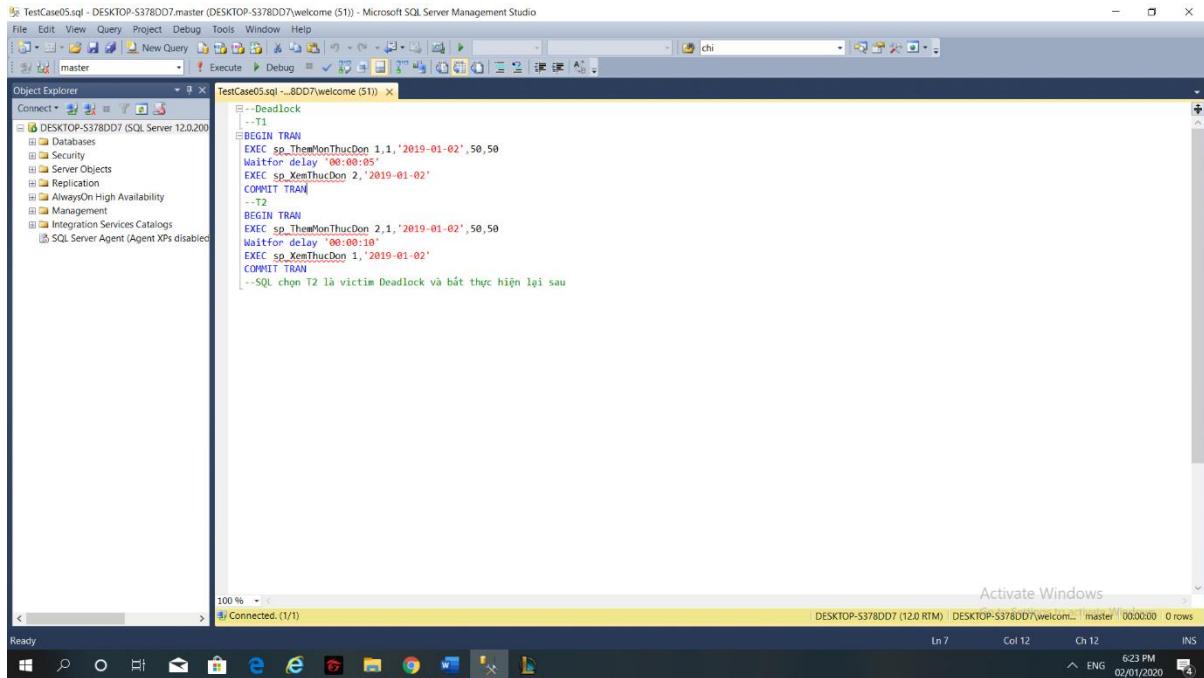
i. Kịch bản lỗi

Giả sử có hai người quản lý công ty, T1 thêm món vào thực đơn chi nhánh 1 và xem thực đơn chi nhánh 2, T2 thêm món vào thực đơn chi nhánh 2 và xem món chi nhánh 1.

T1	T2
BEGIN TRAN EXEC sp_ThemMonThucDon 1,1,'2019-01-02',50,50 Waitfor delay '00:00:05' EXEC sp_XemThucDon 2,'2019-01-02' COMMIT TRAN	BEGIN TRAN EXEC sp_ThemMonThucDon 2,1,'2019-01-02',50,50 Waitfor delay '00:00:10' EXEC sp_XemThucDon 1,'2019-01-02' COMMIT TRAN

ii. Chạy lỗi trên SQL

Đoạn test case của trường hợp Deadlock: (Giả lập lấy cơ chế Read Committed)

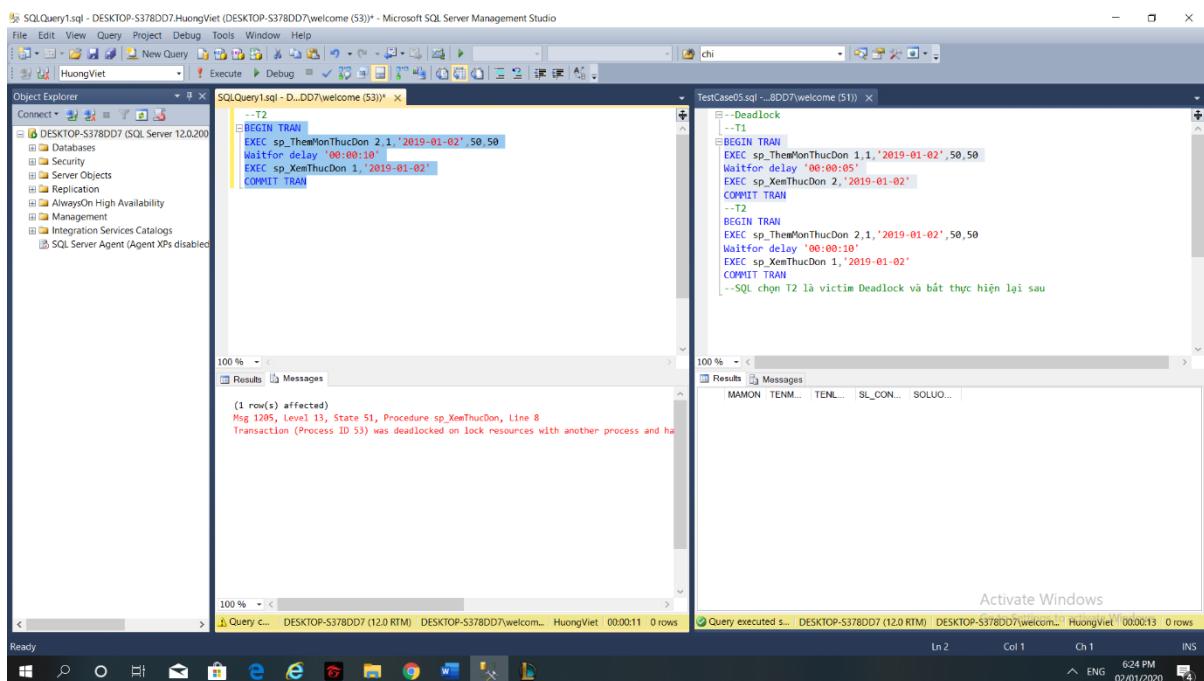


```
--T1
BEGIN TRAN
EXEC sp_ThemMonThucDon 1,1,'2019-01-02',50,50
Waitfor delay '00:00:05'
EXEC sp_XemThucDon 2,'2019-01-02'
COMMIT TRAN
--T2
BEGIN TRAN
EXEC sp_ThemMonThucDon 2,1,'2019-01-02',50,50
Waitfor delay '00:00:10'
EXEC sp_XemThucDon 1,'2019-01-02'
COMMIT TRAN
--SQL chọn T2 là victim Deadlock và bắt thực hiện lại sau
```

Hình 2.5.c.1

Chạy T1 trước rồi chạy T2. Ta thấy chương trình sẽ bị Deadlock vì cả 2 giao tác đều dành khóa ghi trên đơn vị dữ liệu đọc của nhau.

iii. Giải pháp: Cơ chế của SQL khi bị Deadlock



```
--T2
BEGIN TRAN
EXEC sp_ThemMonThucDon 2,1,'2019-01-02',50,50
Waitfor delay '00:00:10'
EXEC sp_XemThucDon 1,'2019-01-02'
COMMIT TRAN
--T1
BEGIN TRAN
EXEC sp_ThemMonThucDon 1,1,'2019-01-02',50,50
Waitfor delay '00:00:05'
EXEC sp_XemThucDon 2,'2019-01-02'
COMMIT TRAN
--SQL chọn T2 là victim Deadlock và bắt thực hiện lại sau
```

Hình 2.5.c.2

T1 được chạy trước, T2 bị SQL SERVER hủy và chạy sau T1 vì chương trình bị DeadLock.

d. *Lỗi d*

❖ **Sinh viên thực hiện:** Trịnh Đức Thanh – 1712769

i. *Kịch bản lỗi*

Nhân viên quản lý khách hàng xem thông tin đơn hàng 3 và sắp cập nhật trạng thái cho đơn hàng 3 từ 'Tiếp nhận' sang 'Đang chuẩn bị', trong lúc đó thành viên 1 muốn hủy đơn hàng 3 đăng nhập vào xem thấy trạng thái vẫn là 'Tiếp nhận' và thực hiện hủy đơn => Deadlock xảy ra.

Ở đây, ta set cơ chế khóa Serializable cho cả T1 và T2.

T1	T2
BEGIN TRAN SET TRAN ISOLATION LEVEL SERIALIZABLE SELECT*FROM DONHANG WHERE MADH=3 waitfor delay '00:00:20'	BEGIN TRAN SET TRAN ISOLATION LEVEL SERIALIZABLE SELECT*FROM DONHANG WHERE MADH=3
UPDATE DONHANG SET TRANGTHAI=N'Đang chuẩn bị' WHERE MADH=3	UPDATE DONHANG SET HUYDON=1, TRANGTHAI=N'Đã hủy'



ii. Chạy lỗi trên SQL

Chạy đoạn test case gây lỗi như hình 2.5.d.1:

```

--Lỗi 5: Deadlock
BEGIN TRAN
SET TRAN ISOLATION LEVEL SERIALIZABLE
SELECT * FROM DONHANG
WHERE MADH=3
waitfor delay '00:00:20'
UPDATE DONHANG
SET TRANGTHAI=N'Dang chuẩn bị'
WHERE MADH=3
COMMIT
    
```

MADH	MANV	MATV	MACN	NGAYDAT	TRANGTHAI	KENHDH	THANHTIEN	TONGTIEN	PTTT	
1	3	2	1	1	2019-10-27	Tiếp nhận	Điện thoại	380000	430000	NULL


```

--Lỗi 5: Deadlock
BEGIN TRAN
SET TRAN ISOLATION LEVEL SERIALIZABLE
SELECT * FROM DONHANG
WHERE MADH=3
UPDATE DONHANG
SET HUYDON=1, TRANGTHAI=N'Dã hủy'
WHERE MADH=3
COMMIT
    
```

MADH	MANV	MATV	MACN	NGAYDAT	TRANGTHAI	KENHDH	THANHTIEN	TONGTIEN	PTTT	
1	3	2	1	1	2019-10-27	Tiếp nhận	Điện thoại	380000	430000	NULL

Hình 2.5.d.1

Do set cơ chế khóa là Serializable nên khi T1 đang giữ khóa S, thì T2 vẫn xin được khóa S. Nhưng T2 không xin được khóa X mà phải chờ T1 nhả khóa (commit) và ngược lại T1 cũng không xin được khóa X do phải chờ T2 nhả khóa (commit).

⇒ Deadlock xảy ra.

iii. Giải pháp: Thực hiện rollback giao tác T2

T1	Khóa	T2	Khóa
BEGIN TRAN SET TRAN ISOLATION LEVEL SERIALIZABLE SELECT*FROM DONHANG WHERE MADH=3 waitfor delay '00:00:20'	T1: Xin khóa S SQL: Cấp khóa S T1: Giữ khóa S đến hết giao tác T1		
		BEGIN TRAN SET TRAN ISOLATION LEVEL SERIALIZABLE SELECT*FROM DONHANG WHERE MADH=3	T2: Xin khóa S SQL: Cấp khóa S T2: Giữ khóa S đến hết giao tác T2
UPDATE DONHANG SET TRANGTHAI=N'Đang chuẩn bị' WHERE MADH=3	T1: Xin khóa X T2: ROLLBACK SQL: Cấp khóa X		
COMMIT		BEGIN TRAN SET TRAN ISOLATION LEVEL SERIALIZABLE SELECT*FROM DONHANG WHERE MADH=3	T2: Xin khóa S SQL: Cấp khóa S T2: Giữ khóa S đến hết giao tác T2
		UPDATE DONHANG SET HUYDON=1, TRANGTHAI=N'Dã hủy' WHERE MADH=3	T2: Xin khóa X SQL: Cấp khóa X
		COMMIT	

Ta set cơ chế khóa Serializable cho cả giao tác T1 và T2, lúc này ta thực hiện các giao tác bình thường, T1 xin và được cấp khóa S, T2 cũng xin và được cấp khóa S, sau đó T2 xin khóa X thì không được cấp và phải chờ, sau đó T1 xin khóa X và T2 bị rollback, T1 được cấp khóa X và tiếp tục thực hiện giao tác đến khi commit (Hình 2.5.d.2).

```

51 --Người thực hiện: Trịnh Đức Thành - 1712769
52 BEGIN TRAN
53 SET TRAN ISOLATION LEVEL SERIALIZABLE
54 SELECT * FROM DONHANG
55 WHERE MADH=3
56 waitfor delay '00:00:20'
57 UPDATE DONHANG
58 SET TRANGTHAI=N'Dang chuẩn bị'
59 WHERE MADH=3
60 COMMIT
61
62
63
64
65

```

(1 row(s) affected)
(1 row(s) affected)

```

65
66 --Người thực hiện: Trịnh Đức Thành - 1712769
67
68 --Lỗi 5: Deadlock
69 BEGIN TRAN
70 SET TRAN ISOLATION LEVEL SERIALIZABLE
71 SELECT * FROM DONHANG
72 WHERE MADH=3
73
74 UPDATE DONHANG
75 SET HUVDON=1, TRANGTHAI=N'Dã hủy'
76 WHERE MADH=3
77 COMMIT
78
79

```

(1 row(s) affected)
Msg 1205, Level 13, State 51, Line 74
Transaction (Process ID 52) was deadlocked on lock resources with another process and has been killed.

Hình 2.5.d.2

Sau khi thực hiện xong giao tác T1, ta thực hiện chạy lại giao tác T2. Ta có thể thấy, lúc này trạng thái đơn hàng 3 đã thành ‘Đang chuẩn bị’. Do đó, ta không thực hiện được thao tác hủy đơn (Hình 2.5.d.3).

```

51 --Người thực hiện: Trịnh Đức Thành - 1712769
52 BEGIN TRAN
53 SET TRAN ISOLATION LEVEL SERIALIZABLE
54 SELECT * FROM DONHANG
55 WHERE MADH=3
56 waitfor delay '00:00:20'
57 UPDATE DONHANG
58 SET TRANGTHAI=N'Dang chuẩn bị'
59 WHERE MADH=3
60 COMMIT
61
62
63
64
65

```

(1 row(s) affected)
(1 row(s) affected)

```

65
66 --Người thực hiện: Trịnh Đức Thành - 1712769
67
68 --Lỗi 5: Deadlock
69 BEGIN TRAN
70 SET TRAN ISOLATION LEVEL SERIALIZABLE
71 SELECT * FROM DONHANG
72 WHERE MADH=3
73
74 UPDATE DONHANG
75 SET HUVDON=1, TRANGTHAI=N'Dã hủy'
76 WHERE MADH=3
77 COMMIT
78
79

```

MADH	MANV	MATV	MACN	NGAYDAT	TRANGTHAI	KENHDH	THANTIEN	TONGTIEN	PTT	NUT
1	3	2	1	1	2019-10-27	Đang chuẩn bị	Điện thoại	380000	430000	

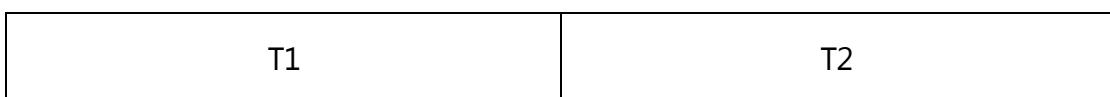
Hình 2.5.d.3

e. Lỗi e

❖ **Sinh viên thực hiện:** Dương Khánh Vi – 1712899

i. Kịch bản lỗi

Lỗi Lost Update ở trên nhưng được giải quyết bằng cơ chế khóa Serializable, dẫn đến Deadlock.



<pre> BEGIN TRAN SET ISOLATION LEVEL SERIALIZABLE SELECT DIACHI FROM THANHVIEN WHERE MATV = 1 waitfor delay'00:00:10' UPDATE THANHVIEN SET DIACHI = N 'Khu đô thị Sala, phường 3, quận 2, TPHCM' COMMIT </pre>	<pre> BEGIN TRAN SET ISOLATION LEVEL SERIALIZABLE SELECT DIACHI FROM THANHVIEN WHERE MATV = 1 UPDATE THANHVIEN SET DIACHI = N' 122, Nguyễn Văn Cừ, phường 10, quận 5, TPHCM' COMMIT </pre>
---	--

ii. Chạy lỗi trên SQL

Chạy đoạn test case của trường hợp Deadlock:

```

TEST CASE LỖI T1...J28QOA\Asus (58)* X
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 3: Lost update
BEGIN TRAN
set tran isolation level SERIALIZABLE
SELECT DIACHI FROM THANHVIEN WHERE MATV = 1
waitfor delay '00:00:10'
UPDATE THANHVIEN
SET DIACHI = N'Khu đô thị Sala, phường 3, quận 2, TPHCM' WHERE MATV = 1
COMMIT

100 % < >
Messages
(1 row(s) affected)

100 % < >
Messages
Msg 1205, Level 13, State 51, Line 6
Transaction (Process ID 59) was deadlocked on lock resources with another process and has been killed.

100 % < >
Messages
Query execut... LAPTOP-NMJ28QOA (12.0 SP1) LAPTOP-NMJ28QOA\Asus (58) HuongViet 00:00:10 0 rows

TEST CASE LỖI T2...J28QOA\Asus (59)* X Giải pháp lỗi T1...M28QOA\Asus (55)
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
BEGIN TRAN
set tran isolation level SERIALIZABLE
SELECT DIACHI FROM THANHVIEN WHERE MATV = 1
UPDATE THANHVIEN
SET DIACHI = N' 122, Nguyễn Văn Cừ, phường 10, quận 5, TPHCM' WHERE MATV = 1
COMMIT

100 % < >
Messages
Msg 1205, Level 13, State 51, Line 6
Transaction (Process ID 59) was deadlocked on lock resources with another process and has been killed.

100 % < >
Messages
Query com... LAPTOP-NMJ28QOA (12.0 SP1) LAPTOP-NMJ28QOA\Asus (59) HuongViet 00:00:04 0 rows

```

Hình 2.5.e.1

iii. Giải pháp

Để giải quyết được Deadlock, Hệ quản trị CSDL đã tự động thực hiện 1 giao tác và hủy bỏ một giao tác còn lại. Cụ thể là thực hiện giao tác T1 và hủy bỏ T2 và yêu cầu người sử dụng chạy lại sau.

T1 commit giao tác:

```

TEST CASE LỖI T1...J28QOA\Asus (58)* X
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 3: Lost update
BEGIN TRAN
set tran isolation level SERIALIZABLE
SELECT DIACHI FROM THANHVIEN WHERE MATV = 1
waitfor delay '00:00:10'
UPDATE THANHVIEN
SET DIACHI = N'Khu đô thị Sala, phường 3, quận 2, TPHCM' WHERE MATV = 1
COMMIT

100 % < >
Messages
Command(s) completed successfully.

100 % < >
Results Messages
DIACHI
1 Khu đô thị Sala, phường 3, quận 2, TPHCM

100 % < >
Messages
Query execut... LAPTOP-NMJ28QOA (12.0 SP1) LAPTOP-NMJ28QOA\Asus (58) HuongViet 00:00:00 0 rows

TEST CASE LỖI T2...J28QOA\Asus (59)* X Giải pháp lỗi T1...M28QOA\Asus (55)
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
BEGIN TRAN
set tran isolation level SERIALIZABLE
SELECT DIACHI FROM THANHVIEN WHERE MATV = 1
UPDATE THANHVIEN
SET DIACHI = N' 122, Nguyễn Văn Cừ, phường 10, quận 5, TPHCM' WHERE MATV = 1
COMMIT

100 % < >
Results Messages
DIACHI
1 Khu đô thị Sala, phường 3, quận 2, TPHCM

100 % < >
Messages
Query ex... LAPTOP-NMJ28QOA (12.0 SP1) LAPTOP-NMJ28QOA\Asus (59) HuongViet 00:00:04 1 rows

```

Hình 2.5.e.2

⇒ T1 sau khi commit, T2 không hiện lỗi deadlock nữa

Thực hiện lại T2:

```

TEST CASE LỖI T1....J28QOA\Asus (58)* X
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
-- Lỗi 3: Lost update
BEGIN TRAN
set tran isolation level SERIALIZABLE
SELECT DIACHI FROM THANHVIEN WHERE MATV = 1
waitfor delay '00:00:10'
UPDATE THANHVIEN
SET DIACHI = N'Khu đô thị Sala, phường 3, quận 2, TPHCM' WHERE MATV = 1
COMMIT

100 % < >
Messages
Command(s) completed successfully.

100 % < >
Query exec... | LAPTOP-NMI28QOA (12.0 SP1) | LAPTOP-NMI28QOA\Asus (58) | HuongViet | 00:00:00 | 0 rows

TEST CASE LỖI T2....J28QOA\Asus (59)* X Giải pháp lỗi T1.s..MJ28QOA\Asus (55)
/*Người thực hiện: Dương Khánh Vi, MSSV: 1712899, Mã nhóm: 8*/
USE HuongViet
BEGIN TRAN
set tran isolation level SERIALIZABLE
SELECT DIACHI FROM THANHVIEN WHERE MATV = 1
UPDATE THANHVIEN
SET DIACHI = N'122, Nguyễn Văn Cử, phường 10, quận 5, TPHCM' WHERE MATV = 1
COMMIT

100 % < >
Results Messages
DIACHI
1 122, Nguyễn Văn Cử, phường 10, quận 5, TPHCM

100 % < >
Query exec... | LAPTOP-NMI28QOA (12.0 SP1) | LAPTOP-NMI28QOA\Asus (59) | HuongViet | 00:00:00 | 1 rows

```

Hình 2.5.e.3

⇒ T2 đã cập nhật dữ liệu thành công.

--HẾT--