

BÁO CÁO ĐỒ ÁN THỰC HÀNH
MÔN HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU
GVHD: Ths Tiết Gia Hồng

THÔNG TIN NHÓM

STT	MSSV	Họ tên	Công việc	% Hoàn thành
1	18120478	Huỳnh Trọng Nghĩa	Unreapeatable Read	100%
2	18120289	Lâm Quốc Bình	Dirty Read	100%
3	19120650	Nguyễn Hoàng Thái	Lost Update	100%
4	18120511	Đào Quang Phúc	Chạy thử code	70%
5	1712820	Bùi Lê Tấn Toàn	Phantom Read	100%


XỬ LÝ TÌNH HUỐNG TRANH CHẤP

I. Sinh viên thực hiện: Nguyễn Hoàng Thái

1. Tình huống 1: Cập nhật giá bán của sản phẩm

ERR01: Lost update T1 (User = đối tác): thực hiện đọc và chỉnh sửa giá của sản phẩm. T2 (User = đối tác): cũng thực hiện đọc và chỉnh sửa giá của sản phẩm trên cùng đơn vị dữ liệu.			
sp_update_gia_Sp	Khóa	sp_update_gia_Sp_2	Khóa
Input: @MaSP, @GiaBan Output: giá bán của sản phẩm được cập nhật		Input: @MaSP, @GiaBan Output: giá bán của sản phẩm được cập nhật	
SET TRAN ISOLATION LEVEL REPEATABLE READ		SET TRAN ISOLATION LEVEL REPEATABLE READ	
BEGIN TRAN			
B1: Kiểm tra thông tin xem có mã sản phẩm đó không IF not exists (select * from sanpham where masanpham = @MaSp) begin Print @MaSP + N'Không tồn tại'; rollback tran; end	S(SanPham)		
B2: Kiểm tra tính hợp lệ của giá bán If @GiaBan <=0 or ISNUMERIC(@GiaBan) !=1 begin Print N'Giá bán nhập vào không hợp lệ'; rollback tran end			


<code>waitfor delay '00:00:08';</code>			
<p>B3: Cập nhật giá sản phẩm vào cửa hàng</p> <pre> update SanPham set GiaBan = @GiaBan where MaSanPham = @MaSP </pre>	<p>X(GiaBan) Khóa exclusive lock được cấp cho GiaBan để ghi dữ liệu, chỉ nhả ra khi hết giao tác.</p>		
		BEGIN TRAN	
		<p>B1: Kiểm tra thông tin xem có mã sản phẩm đó không</p> <pre> IF not exists (select * from sanpham where masanpham = @MaSp) begin Print @MaSP + N'Không tồn tại'; rollback tran; end </pre>	S(SanPham)
		<p>B2: Kiểm tra tính hợp lệ của giá bán</p> <pre> If @GiaBan <=0 or ISNUMERIC(@GiaBan) !=1 begin Print N'Giá bán nhập vào không hợp lệ'; rollback tran end </pre>	
		<code>waitfor delay '00:00:01';</code>	

		B3: Cập nhật giá sản phẩm vào cửa hàng <code>update SanPham</code> <code>set GiaBan = @GiaBan</code> <code>where MaSanPham = @MaSP</code>	Không được cấp khóa vì transaction 1 đã giữ  không ghi được
COMMIT		COMMIT	

2. Tình huống 2: Cập nhật thời gian hiệu lực hợp đồng trong bảng Hợp Đồng

ERR01: Lost update T1 (User = nhân viên): thực hiện đọc và chỉnh sửa thời gian hiệu lực của hợp đồng. T2 (User = nhân viên): cũng thực hiện đọc và chỉnh sửa thời gian hiệu lực của hợp đồng trên cùng đơn vị dữ liệu.			
sp_update_TGHieuLuc_HOPDONG	Khóa	sp_update_TGHieuLuc_HOPDONG_2	Khóa
Input: @MaHD int, @NgayHieuLuc date Output: Ngày hiệu lực của hợp đồng được cập nhật		Input: @MaHD int, @NgayHieuLuc date Output: Ngày hiệu lực của hợp đồng được cập nhật	
SET TRAN ISOLATION LEVEL REPEATABLE READ		SET TRAN ISOLATION LEVEL REPEATABLE READ	
BEGIN TRAN			
B1: Kiểm tra tính hợp lệ của mã hợp đồng if not exists (select * from HOPDONG where MaHD =@MAHD) begin print N'Mã hợp đồng không tồn tại'; rollback tran; end	S(HopDong)		
B2: Kiểm tra tính hợp lệ của chuỗi nhập vào tình trạng if ISDATE(cast(@NgayHieuLuc as nvarchar)) != 1 begin print N'Ngày nhập vào không hợp lệ'; rollback tran ; end			
B3: Kiểm tra ngày hiệu lực có nhỏ hơn ngày hết hạn	S(TGHetHan)		

<pre> declare @NgayHetHan date; select @NgayHetHan = TGHetHan from HOPDONG where @MAHD = MaHD if @NgayHieuLuc > @NgayHetHan begin print N'Ngày hiệu lực phải nhỏ hơn ngày hết hạn'; rollback tran; end </pre>			
<pre> waitfor delay '00:00:06'; </pre>			
<p>B4: Cập nhật ngày hiệu lực trong hợp đồng</p> <pre> update HOPDONG set TGHIEULUC = @NgayHieuLuc where MaHD = @MAHD </pre>	<p>Khóa exclusive lock được cấp cho NgayHieuLuc để ghi dữ liệu, chỉ nhả ra khi hết giao tác</p>		
		BEGIN TRAN	
		<p>B1: Kiểm tra tính hợp lệ của mã hợp đồng</p> <pre> if not exists (select * from HOPDONG where MaHD = @MAHD) begin print N'Mã hợp đồng không tồn tại'; </pre>	S(HopDong)


		<pre>rollback tran; end</pre>	
		<p>B2: Kiểm tra tính hợp lệ của chuỗi nhập vào tình trạng</p> <pre>if ISDATE(cast(@NgayHieuLuc as nvarchar)) != 1 begin print N'Ngày nhập vào không hợp lệ'; rollback tran ; end</pre>	
		<p>B3: Kiểm tra ngày hiệu lực có nhỏ hơn ngày hết hạn</p> <pre>declare @NgayHetHan date; select @NgayHetHan = TGHetHan from HOPDONG where @MAHD = MaHD if @NgayHieuLuc > @NgayHetHan begin print N'Ngày hiệu lực phải nhỏ hơn ngày hết hạn'; rollback tran; end</pre>	S(TGHetHan)
		<pre>waitfor delay '00:00:01';</pre>	
		<p>B4: Cập nhật ngày hiệu lực trong hợp đồng</p> <pre>update HOPDONG set TGHIEULUC = @NgayHieuLuc where MaHD = @MAHD</pre>	Không được cấp khóa vì transaction 1 đã giữ  không ghi được

COMMIT		COMMIT	
--------	--	--------	--

3. Tình huống 3: Cập nhật tình trạng đặt hàng trong bảng PHIEUGIAOHANG

ERR01: Lost update T1 (User = Tài xế): thực hiện đọc và chỉnh sửa tình trạng đơn hàng T2 (User = Tài xế): cũng thực hiện đọc và chỉnh sửa tình trạng đơn hàng trên cùng đơn vị dữ liệu			
sp_update_TinhTrangDatHang	Khóa	sp_update_TinhTrangDatHang_2	Khóa
Input: @MaPG int, --Mã Phiếu Giao @TinhTrang nvarchar(20) -Tình trạng giao hàng Output: tình trạng giao hàng được cập nhật lại		Input: @MaPG int, --Mã Phiếu Giao @TinhTrang nvarchar(20) -Tình trạng giao hàng Output: tình trạng giao hàng được cập nhật lại	
SET TRAN ISOLATION LEVEL REPEATABLE READ		SET TRAN ISOLATION LEVEL REPEATABLE READ	
BEGIN TRAN			
B1: Kiểm tra tính hợp lệ của mã phiếu giao if not exists(select * from PHIEUGIAOHANG where @MaPG = MaPhieuGiao) begin print N'Mã phiếu giao không hợp lệ'; rollback tran; end	S(PhieuGiaoHang)		
B2: Kiểm tra tình trạng nhập vào có phải là chuỗi hay không if ISNUMERIC(@TinhTrang) = 1 begin			

<pre>print N'Tình trạng nhập không phải là chuỗi'; rollback tran; end</pre>			
<pre>waitfor delay '00:00:08';</pre>			
<pre>B3: Cập nhật giá sản phẩm vào cửa hàng update PHIEUGIAOHANG set TinhTrangGiaoHang= @TinhTrang where MaPhieuGiao = @MaPG</pre>	Khóa exclusive lock được cấp cho TinhTrangGiaoHang để ghi dữ liệu, chỉ nhả ra khi hết giao tác		
		BEGIN TRAN	
		<pre>B1: Kiểm tra tính hợp lệ của mã phiếu giao if not exists(select * from PHIEUGIAOHANG where @MaPG = MaPhieuGiao) begin print N'Mã phiếu giao không hợp lệ'; rollback tran; end</pre>	S(PhieuGiaoH ang)
		<pre>B2: Kiểm tra tình trạng nhập vào có phải là chuỗi hay không if ISNUMERIC(@TinhTrang) = 1 begin print N'Tình trạng nhập không phải là chuỗi';</pre>	

		<code>rollback tran;</code>	
		<code>end</code>	
		<code>waitfor delay '00:00:01';</code>	
		B3: Cập nhật giá sản phẩm vào cửa hàng <code>update</code> PHIEUGIAOHANG <code>set</code> TinhTrangGiaoHang= @TinhTrang <code>where</code> MaPhieuGiao = @MaPG	Không được cấp khóa vì transaction 1 đã giữ  không ghi được
COMMIT		COMMIT	

4. Tình huống 4: Cập nhật ngày cung cấp lại trong bảng CUNGCAPSANPHAM

ERR01: Lost update T1 (User = đối tác): thực hiện đọc và chỉnh sửa ngày cung cấp của sản phẩm. T2 (User = đối tác): cũng thực hiện đọc và chỉnh mà ngày cung cấp trên cùng đơn vị dữ liệu			
sp_update_NgayCC_CCSP	Khóa	sp_update_NgayCC_CCSP_2	Khóa
Input: @MaSP, @MaCN, @NgayCC Output: ngày cung cấp của sản phẩm được cập nhật		Input: @MaSP, @MaCN, @NgayCC Output: ngày cung cấp của sản phẩm được cập nhật	
SET TRAN ISOLATION LEVEL REPEATABLE READ		SET TRAN ISOLATION LEVEL REPEATABLE READ	
BEGIN TRAN			
B1: Kiểm tra tính hợp lệ của mã sản phẩm <code>if not exists (select * from CUNGCAPSANPHAM</code> <code>where @MaSP = MaSanPham and @MACN =</code> <code>MaChiNhanh)</code>	S(CUNGCAPSA NPHAM)		

<pre> begin print N'Không tồn tại mã sản phẩm' rollback tran end </pre>			
<pre> B2: Kiểm tra kiểu dữ liệu của mã chi nhánh if ISNUMERIC(@MACN) !=1 begin print N'Không đúng kiểu dữ liệu cho mã chi nhánh' rollback tran; end </pre>			
<pre> waitfor delay '00:00:08'; </pre>			
<pre> B3: Cập nhật giá sản phẩm vào cửa hàng update CUNGCAPSANPHAM set NgayCC = @NgayCC where MaSanPham = @MaSP and MaChiNhanh = @MaCN </pre>	<p>Khóa exclusive lock được cấp cho NgayCC để ghi dữ liệu, chỉ nhả ra khi hết giao tác</p>		
Commit tran		BEGIN TRAN	
		<pre> B1: Kiểm tra tính hợp lệ của mã sản phẩm if not exists (select * from CUNGCAPSANPHAM where @MaSP = MaSanPham and @MACN = MaChiNhanh) begin </pre>	S(CUNGCAPSANPHAM)

		<pre> print N'Không tồn tại mã sản phẩm' rollback tran end </pre>	
		<pre> B2: Kiểm tra kiểu dữ liệu của mã chi nhánh if ISNUMERIC(@MACN) !=1 begin print N'Không đúng kiểu dữ liệu cho mã chi nhánh' rollback tran; end </pre>	
		<pre> waitfor delay '00:00:08'; </pre>	
		<pre> B3: Cập nhật giá sản phẩm vào cửa hàng update CUNGCAPSANPHAM set NgayCC = @NgayCC where MaSanPham = @MaSP and MaChiNhanh = @MaCN </pre>	Không được cấp khóa vì transaction 1 đã giữ ? không ghi được
COMMIT		COMMIT	

II. Sinh viên thực hiện: Huỳnh Trọng Nghĩa

1. Tình huống 1: Thay đổi giá sản phẩm trong khi đang đọc

ERR01: Unrepeatable Read T1 (User = khách hàng): Xem giá sản phẩm có mã sản phẩm là 1 T2 (User = đối tác): thay đổi giá sản phẩm có mã sản phẩm là 1 từ 12000 thành 20000			
sp_XemGiaSP	Khóa	sp_ChinhSuaGiaSP	Khóa
Input: @inputMaSanPham = '23' Output1: Gia của san pham la 12,000 Output2: Gia của san pham la 20,000		Input: @inputMaSanPham = '23' @inputGiaBan = '20000' Output: Chinh sua gia thanh cong	
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ		SET TRANSACTION ISOLATION LEVEL REPEATABLE READ	
BEGIN TRAN		BEGIN TRAN	
B1: Kiểm tra thông tin (1) <pre> if exists (select SP.GiaBan from SanPham as SP where SP.MaSanPham = @inputMaSanPham) </pre>	S(SanPham) //Xin khoá đọc trên bảng SanPham, sau bước này không nhả khoá S	B1: Kiểm tra thông tin <pre> if exists (select * from SanPham as SP where SP.MaSanPham = @inputMaSanPham) </pre>	S(SanPham) //Xin khoá đọc trên bảng SanPham, sau bước này không nhả khoá S
B2: Xem giá sản phẩm <pre> begin print('Gia của san pham la') select SP.GiaBan </pre>			

<pre> from SanPham as SP where SP.MaSanPham = <input type="text"/> waitfor delay '00:00:08'; </pre>			
		<p>B2: Chỉnh sửa giá sản phẩm</p> <pre> begin Update SanPham Set GiaBan = <input type="text"/> where MaSanPham = <input type="text"/> end else begin print('San Pham khong ton tai'); end end </pre>	<p>Tại đây DBMS giữ khóa S nên không thể cấp khóa X được, phải chờ T1 commit thì T2 mới chạy tiếp được</p>
<p>B3: Xem lại giá sản phẩm</p> <pre> select SP.GiaBan </pre>			

<pre> from SanPham as SP where SP.MaSanPham = <input type="text"/> end else begin print('San pham khong ton tai') end </pre>			
<pre> commit transaction; END </pre>	Nhả khoá S		Sau khi T1 commit và nhả khoá S thì mới cấp khoá X(SanPham) được
		<pre> commit transaction; END </pre>	Nhả khoá X
Không bị lỗi nữa, T1 phải xong hoàn toàn thì T2 mới bắt đầu được write dữ liệu			

2. Tình huống 2: Huỷ đơn hàng

ERR02: Unrepeatable Read T1 (User = Tài xế): Tài xế đang xem thông tin đơn hàng có mã là '12' để nhận đơn hàng giao đi T2 (User = Khách hàng): huỷ đơn hàng số '12'			
sp_XemDonHang	Khóa	sp_HuyDonHang	Khóa
Input: @inputMaDonHang = '12' Output1: Thông tin đơn hàng số 12 Output2: 'Don hang khong ton tai'		Input: @requestMaKhachHang = '258' @requestMaDonHang = '12' Output: 'Huy don hang thanh cong'	
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ		SET TRANSACTION ISOLATION LEVEL REPEATABLE READ	
BEGIN TRAN		BEGIN TRAN	
B1: Kiểm tra thông tin (1) <pre>if exists (select * from DonDatHang as DDH where DDH.MaDonHang = @inputMaDonHang)</pre>	S(DonDatHang) //Xin khoá đọc trên bảng DonDatHang, sau bước này không nhả khoá S	B1: Kiểm tra thông tin (1) <pre>if exists (select * from DonDatHang as DDH where DDH.MaDonHang = @requestMaDonHang and DDH.MaKhachHang = @requestMaKH)</pre>	S(DonDatHang) Xin khoá đọc trên bảng DonDatHang, sau bước này nhả khoá S
B2: Xem đơn hàng <pre>print('Chi Tiet Don Hang') select * from DonDatHang as DDH where DDH.MaDonHang = @inputMaDonHang</pre>			

<pre>waitfor delay '00:00:08';</pre>			
		<p>B2: Huỷ đơn hàng</p> <pre>begin Delete from ChiTietDonDatHang where MaDonHang = @requestMaDonHang Delete from DonDatHang where MaDonHang = @requestMaDonHang and MaKhachHang = @requestMaKH print('Huy don hang thanh cong') end else begin print('Ma don hang khong hop le') end</pre>	<p>Không cấp X(DonDatHang) được vì khoá S chưa được nhả nên phải chờ T1 commit</p>

B3: Xem lại đơn hàng			
<pre>print('Chi Tiết Đơn Hàng') select * from DonDatHang as DDH where DDH.MaDonHang = <input type="text" value="@inputMaDonHang"/></pre>			
<pre>commit tran</pre>	Nhả khoá S		Sau khi T1 commit mới cấp khóa X(DonDatHang)
		<pre>commit tran</pre>	Nhả khoá X
Không lỗi nữa			

3. Tình huống 3: Chỉnh sửa địa chỉ của chi nhánh

ERR03: Unrepeatable Read T1 (User = tài xế): Tài xế đã nhận đơn hàng A và đang xem địa chỉ chi nhánh tương ứng để đi lấy hàng T2 (User = đối tác): Chỉnh sửa địa chỉ chi nhánh từ 65 Phan Đình Phùng, Phường 3, Quận 12, TPHCM thành 53 Phạm Thế Hiền, Phường 8, Quận 12, TPHCM			
sp_XemChiNhanh	Khóa	sp_ChinhSuaDiaChiCN	Khóa
<u>Input:</u> @inputMaChiNhanh = '18' <u>Output1:</u> 65 Phan Đình Phùng, Phường 3, Quận 12, TPHCM		<u>Input:</u> @requestMaChiNhanh = '23' @requestMaDoitac = '15' @DiaChiMoi = '53 Phạm Thế Hiền, Phường 8, Quận 12, TPHCM'	

Output2: 53 Pham The Hien, Phuong 8, Quan 12, TPHCM		Output: Chinh sua dia chi chi nhanh thanh cong	
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ		SET TRANSACTION ISOLATION LEVEL REPEATABLE READ	
BEGIN TRAN		BEGIN TRAN	
B1: Kiểm tra thông tin (1) <pre>if exists (select * from ChiNhanh as CN where CN.MaChiNhanh = @inputMaChiNhanh)</pre>	S(ChiNhanh) //Xin khoá đọc trên bảng ChiNhanh, Sau bước này không nhả khóa S	B1: Kiểm tra thông tin <pre>if exists (select * from ChiNhanh as CN where CN.MaChiNhanh = @requestMaChiNhanh and CN.MaDoiTac = @requestMaDoiTac)</pre>	S(ChiNhanh) //Xin khoá đọc trên bảng ChiNhanh, sau bước này không nhả khóa S
B2: Xem địa chỉ <pre>begin print('Dia chi') select CN.DiaChiChiNhanh from ChiNhanh as CN where CN.MaChiNhanh = @inputMaChiNhanh waitfor delay '00:00:08';</pre>			

		B2: Chỉnh sửa địa chỉ <pre> begin Update ChiNhanh set DiaChiChiNhanh = @DiaChiMoi where MaChiNhanh = @requestMaChiNhanh print('Thay doi dia chi thanh cong') end else begin print('Thay doi dia chi khong thanh cong') end </pre>	Không cấp X(ChiNhanh) được vì khoá S chưa được nhả nên phải chờ T1 commit
B3: Xem lại địa chỉ <pre> print('Dia chi') select CN.DiaChiChiNhanh from ChiNhanh as CN where CN.MaChiNhanh = @inputMaChiNhanh end else begin </pre>			

<pre>print('Khong tim thay dia chi hop le') end</pre>			
<pre>commit tran</pre>		<pre>commit tran END</pre>	Sau khi T1 commit mới cấp khoá S được
			Nhả khoá X
Không lỗi nữa			

4. Tình huống 4: duyệt hợp đồng khi người khác đang xem

ERR03: Unrepeatable Read T1 (User = nhân viên 1): Nhân viên 1 xem trạng thái duyệt một hợp đồng T2 (User = nhân viên 2): Duyệt hợp đồng			
sp_XemHopDong	Khóa	sp_DuyetHopDong	Khóa
<u>Input:</u> @inputMaHopDong = '18' <u>Output1:</u> Thông tin Hop dong Chưa duyệt <u>Output2:</u> Thông tin Hop dong đã duyệt		<u>Input:</u> @requestMaHopDong = '18' <u>Output:</u> 'Duyệt Hop dong thanh cong'	
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ		SET TRANSACTION ISOLATION LEVEL REPEATABLE READ	
BEGIN TRAN		BEGIN TRAN	
B1: Kiểm tra thông tin (1)	S(HopDong)	B1: Kiểm tra thông tin	S(HopDong)

<pre>if exists (select * from GiaHanHopDong as GHHD where GHHD.MAHD = <inputmahopdong>)</inputmahopdong></pre>	//Xin khoá đọc trên bảng ChiNhanh, Sau bước này không nhả khóa S	<pre>if exists (select * from GiaHanHopDong as GHHD where GHHD.MaHD = <inputmahopdong>)</inputmahopdong></pre>	//Xin khoá đọc trên bảng ChiNhanh, sau bước này không nhả khóa S
<p>B2: Xem hợp đồng</p> <pre>begin print('Thông tin Hợp đồng') select GHHD.TinhTrangGiaHan from GiaHanHopDong as GHHD where GHHD.MAHD = <inputmahopdong '00:00:08';<="" delay="" pre="" waitfor=""> </inputmahopdong></pre>			
		<p>B2: Duyệt hợp đồng</p> <pre>begin Update GiaHanHopDong set TinhTrangGiaHan = 'daduyet' where GiaHanHopDong.MAHD = <inputmahopdong< pre=""> </inputmahopdong<></pre>	<p>Không cấp X(HopDong) được vì khoá S chưa được nhả nên phải chờ T1 commit</p>

		<pre> set @thoiGianGiaHan = (select GHHD.ThoiGianGiaHan fro m GiaHanHopDong as GHHD whe re GHHD.MAHD = @requestMaHopDong) Update HopDong set TGHetHan = @thoiGianGiaHan where MaHD = @requestMaHopDong print('Duyet Hop dong thanh cong') end else begin print('Duyet hop dong khong thanh cong') end</pre>	
--	--	--	--

<p>B3: Xem lại thông tin hợp đồng</p> <pre>print('Thông tin Hop dong') select GHHD.TinhTrangGiaHan from GiaHanHopDong as GHHD where GHHD.MAHD = @inputMaHopDong end else begin print('Khong tim thay Hop Dong') rollback tran end</pre>			
<pre>commit tran</pre>			
	Nhả khoá S	<pre>commit tran</pre>	Sau khi T1 commit thì cấp khoá X
Không lỗi nữa			Nhả khoá X

III. Sinh viên thực hiện: Lâm Quốc Bình

1. Tình huống 1: Khách hàng đang cập nhật địa chỉ giao hàng nhưng do địa chỉ nhập vào quá dài hệ thống báo lỗi nên phải rollback. Cùng lúc đó tài xế xem địa chỉ giao hàng.

ERR01: Dirty read T1 (User = Khách hàng): cập nhật địa chỉ giao hàng. T2 (User = Tài xế): xem địa chỉ giao hàng			
KH_Update_DCGH	Khóa	TX_View_DCGH	Khóa
Input: @MaDH, @DCGH Output: Thông tin đơn hàng đã cập nhật		Input: @MaDH Output: Thông tin đơn hàng	
SET TRANSACTION ISOLATION LEVEL READ COMMITTED		SET TRANSACTION ISOLATION LEVEL READ COMMITTED	
BEGIN TRAN			
B1: Kiểm tra mã đơn hàng IF NOT EXISTS (SELECT * FROM DonDatHang WHERE MaDonHang = @MaDH) BEGIN PRINT (N'Mã Đơn hàng không tồn tại'); ROLLBACK TRAN; END	S(DonDatHang)		
B2: Update địa chỉ giao hàng UPDATE DonDatHang SET DiaChiGiaoHang = @DCGH WHERE MaDonHang = @MaDH	X(DonDatHang)		

		BEGIN TRAN	
		B1: Kiểm tra mã đơn hàng IF NOT EXISTS (SELECT * FROM DonDatHang WHERE MaDonHang = @MaDH) BEGIN PRINT (N'Mã Đơn hàng không tồn tại'); ROLLBACK TRAN; END	S(DonDatHang)
		B2: Xem thông tin đơn hàng SELECT * FROM DonDatHang WHERE MaDonHang = @MaDH	S(DonDatHang)
IF (LEN(@DCGH) > 150) BEGIN PRINT (N'Địa chỉ giao hàng quá dài'); WAITFOR DELAY '00:00:08'; ROLLBACK TRAN; END			
COMMIT TRAN		COMMIT TRAN	

2. Tình huống 2: Đối tác cập nhật giá sản phẩm nhưng do giá sản phẩm nhập vào có kiểu dữ liệu không phù hợp nên phải rollback. Cùng lúc đó khách hàng xem giá sản phẩm.

ERR02: Dirty read T1 (User = Đối tác): cập nhật giá sản phẩm. T2 (User = Khách hàng): xem giá sản phẩm			
DT_Update_GiaSP	Khóa	KH_View_SP	Khóa
Input: @MaSP, @GiaSP		Input: @MaSP	
Output: Thông tin sản phẩm đã cập nhật		Output: Thông tin sản phẩm	
SET TRANSACTION ISOLATION LEVEL READ COMMITTED		SET TRANSACTION ISOLATION LEVEL READ COMMITTED	
BEGIN TRAN			
B1: Kiểm tra mã sản phẩm IF NOT EXISTS (SELECT * FROM SanPham where MaSanPham = @MaSP) BEGIN PRINT @MaSP + N'Không tồn tại'; ROLLBACK TRAN; END	S(SanPham)		
B2: Update giá sản phẩm UPDATE SanPham set GiaBan = @GiaBan where MaSanPham = @MaSP	X(SanPham)		
		BEGIN TRAN	
		B1: Kiểm tra mã sản phẩm	S(SanPham)

		<pre> IF NOT EXISTS (SELECT * FROM SanPham WHERE MaSanPham = @MaSp) BEGIN PRINT @MaSp + N'Không tồn tại' ROLLBACK TRAN; END </pre>	
		<p>B2: Xem thông tin đơn hàng</p> <pre> SELECT * FROM SanPham WHERE MaSanPham = @MaSp </pre>	S(SanPham)
<pre> If @GiaBan <=0 or ISNUMERIC(@GiaBan) !=1 BEGIN PRINT N'Giá bán nhập vào không hợp lệ'; WAITFOR DELAY '00:00:08'; ROLLBACK TRAN; end </pre>			
COMMIT TRAN		COMMIT TRAN	

3. Tình huống 3: Khách hàng đã đặt hàng nhưng lại thay đổi địa chỉ giao hàng và địa chỉ giao hàng nhập vào quá ngắn nên phải rollback. Cùng lúc đó tài xế xem thông tin của đơn đặt hàng.

ERR01: Dirty read T1 (User = Khách hàng): cập nhật địa chỉ giao hàng. T2 (User = Tài xế): xem địa chỉ giao hàng			
KH_Update_DCGH	Khóa	TX_View_DCGH	Khóa
<p>Input: @MaDH, @DCGH</p> <p>Output: Thông tin đơn hàng đã cập nhật</p>		<p>Input: @MaDH</p> <p>Output: Thông tin đơn hàng</p>	

SET TRANSACTION ISOLATION LEVEL READ COMMITTED		SET TRANSACTION ISOLATION LEVEL READ COMMITTED	
BEGIN TRAN			
B1: Kiểm tra mã đơn hàng IF NOT EXISTS (SELECT * FROM DonDatHang WHERE MaDonHang = @MaDH) BEGIN PRINT (N'Mã Đơn hàng không tồn tại'); ROLLBACK TRAN; END	S(DonDatHang)		
B2: Update địa chỉ giao hàng UPDATE DonDatHang SET DiaChiGiaoHang = @DCGH WHERE MaDonHang = @MaDH	X(DonDatHang)		
		BEGIN TRAN	
		B1: Kiểm tra mã đơn hàng IF NOT EXISTS (SELECT * FROM DonDatHang WHERE MaDonHang = @MaDH) BEGIN PRINT (N'Mã Đơn hàng không tồn tại'); ROLLBACK TRAN; END	S(DonDatHang)
		B2: Xem thông tin đơn hàng SELECT * FROM DonDatHang	S(DonDatHang)

		WHERE MaDonHang = @MaDH	
<pre> IF (LEN(@DCGH) < 30) BEGIN PRINT (N'Địa chỉ giao hàng quá ngắn'); WAITFOR DELAY '00:00:08'; ROLLBACK TRAN; END </pre>			
COMMIT TRAN		COMMIT TRAN	

4. Tình huống 4: Nhân viên gia hạn hợp đồng nhưng khi nhập thời gian hết hạn hợp đồng không hợp lệ nên phải rollback. Cùng lúc đó đối tác vào xem hợp đồng đã gia hạn chưa

ERR02: Dirty read T1 (User = Nhân viên): gia hạn hợp đồng T2 (User = Đối tác): xem hợp đồng			
NV_Update_TGGH	Khóa	DT_View_HD	Khóa
<u>Input:</u> @MaHD, @TGGH <u>Output:</u> Thông tin hợp đồng đã được cập nhật		<u>Input:</u> @MaHD <u>Output:</u> Thông tin hợp đồng	
SET TRANSACTION ISOLATION LEVEL READ COMMITTED		SET TRANSACTION ISOLATION LEVEL READ COMMITTED	
BEGIN TRAN			
B1: Kiểm tra mã hợp đồng	S(GIAHANHOPDONG)		

<pre>IF NOT EXISTS (SELECT * FROM GiaHanHopDong WHERE MAHD = @MaHD) BEGIN PRINT (N'Mã hợp đồng không tồn tại'); ROLLBACK TRAN; END</pre>			
<p>B2: Update giá sản phẩm</p> <pre>UPDATE GiaHanHopDong SET ThoiGianGiaHan = @TGGH WHERE MAHD = @MaHD</pre>	X(GIAHANHOPDONG)		
		BEGIN TRAN	
		<p>B1: Kiểm tra mã hợp đồng</p> <pre>IF NOT EXISTS (SELECT * FROM GiaHanHopDong WHERE MAHD = @MaHD) BEGIN PRINT (N'Mã hợp đồng không tồn tại'); ROLLBACK TRAN; END</pre>	S(GIAHANHOPDONG)
		<p>B2: Xem thông tin đơn hàng</p> <pre>SELECT * FROM GiaHanHopDong WHERE MAHD = @MaHD</pre>	S(GIAHANHOPDONG)
<pre>IF ISDATE(@TGGH) != 1 BEGIN PRINT N'Thời gian gia hạn không hợp lệ';</pre>			

ROLLBACK TRAN; END			
COMMIT TRAN		COMMIT TRAN	

IV. Sinh viên thực hiện: Bùi Lê Tấn Toàn

1. Tình huống 1: Xem thông tin sản phẩm của đối tác

ERR01: Phantom read T1 (User = Khách hàng): thực hiện xem thông tin các sản phẩm của 1 đối tác. T2 (User = Đối tác): thực hiện thêm 1 sản phẩm vào danh sách sản phẩm của đối tác			
sp_XemSP	Khóa	sp_SPMoi	Khóa
Input: @MaDT Output: Danh sách các sản phẩm		Input: @MaSP, @MaCC, @TenSP, @GiaBan, @Phanloai, @GhiChu Output: Thông tin sản phẩm mới	
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE		SET TRANSACTION ISOLATION LEVEL READ COMMITTED	
BEGIN TRAN			
B1: Kiểm tra thông tin MaDT set tran ISOLATION LEVEL READ COMMITTED IF not exists (select * from DoiTac where MaDoiTac = @MaDT) begin Print @MaDT + N'Không tồn tại';	S(DoiTac) //Xin khoá đọc trên bảng Doi Tac. sau bước này nhả khoá S		

rollback tran; end			
B2: Xem thông tin sản phẩm của @MaDT SELECT s.* FROM DoiTac d, SanPham s WHERE d.MaDoiTac=@MaDT and d.MaDoiTac=s.MaDoiTac	S(SanPham, DoiTac, CungCapSanPham , ChiNhanh) //Xin khoá đọc trên bảng SanPham, DoiTac, sau đó nhả khóa S		
WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	
		B1: thêm thông tin vào bảng sản phẩm INSERT dbo.SanPham (MaSanPham, TenSanPham, GiaBan, PhanLoaiHang, MaDoiTac) VALUES (@MaSP, @TenSP, @Giaban, @PLHang,	X(SanPham)

		@MaDT)	
		<p>B2: thêm thông tin vào bảng Cung cấp sản phẩm</p> <pre> INSERT CungCapSanPham (MaChiNhanh, MaSanPham, GhiChu, NgayCC) Values (@MaCC, @MaSP, @GhiChu, @NgayCC) </pre>	X(CungCapSanPham)
		COMMIT	
<p>B2. Xem lần 2 danh sách sản phẩm</p> <pre> SELECT s.* FROM DoiTac d, SanPham s WHERE d.MaDoiTac=@MaDT and d.MaDoiTac=s.MaDoiTac </pre>	<p>S(SanPham, DoiTac, CungCapSanPham, ChiNhanh) //Xin khoá đọc trên bảng SanPham, DoiTac</p>		
COMMIT TRAN			

Thao tác xem thông tin 2 lần thì ở mức LEVEL SERIALIZABLE thì T1 thực thi xong mới chạy T2 nên không xảy ra tình trạng Phantom read			
--	--	--	--

2. Tình huống 2: Xem danh sách hợp đồng

ERR01: Phantom read T1 (User = Đối Tác): thực hiện xem danh sách các hợp đồng T2 (User = Nhân viên): thực hiện đăng ký 1 hợp đồng mới			
sp_XemHD	Khóa	sp_HDMoi	Khóa
Input: Output: Danh sách các hợp đồng		Input: @Mathue, @MaDT, @SoCN, @Phi, @TGHL, @TGHH Output: Thông tin Hợp đồng mới	
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE		SET TRANSACTION ISOLATION LEVEL READ COMMITTED	
BEGIN TRAN			
B1. Xem lần 1 danh sách hợp đồng SELECT h.* FROM HopDong h	S(HopDong) //Xin khoá đọc trên bảng HopDong sau đó nhả khoá		
WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	
		if not exists (select * from ChiNhanh where @MaDT = MaDoiTac and @SoCN = MaChiNhanh) begin	

		<pre> raiserror(N'Không tồn tại chi nhánh của đối tác này',16,1) rollback; end </pre>	
		<pre> declare @MaHD int; select @MaHD = Max(MaHD) from HopDong set @MaHD = @MaHD +1; </pre>	
		<pre> B1: thêm thông tin vào bảng sản phẩm INSERT dbo.HopDong (MaHD, MaSoThue, MaDoiTac, SoChiNhanh, PhiKichHoat, TGHieuLuc, TGHetHan, GiaHan) VALUES (@Mahd, @Mathue, @MaDT, @SoCN, @Phi, @TGHL, </pre>	X(HopDong)

		@TGHH, 1)	
		COMMIT	
B2. Xem lần 2 danh sách hợp đồng SELECT h.* FROM HopDong h COMMIT TRAN	S(HopDong)		
Thao tác xem thông tin 2 lần thì ở mức LEVEL SERIALIZABLE thì T1 thực thi xong mới chạy T2 nên không xảy ra tình trạng Phantom read			

3. Tình huống 3: Xem các đơn đặt hàng

ERR01: Phantom read T1 (User = Tài xế): thực hiện xem danh sách các đơn hàng T2 (User = Khách Hàng): thực hiện thêm 1 đơn hàng mới			
sp_XemDH	Khóa	sp_DHMoi	Khóa
<u>Input:</u> <u>Output:</u> Danh sách các đơn hàng		<u>Input:</u> @MaDH, @MaKH, @NgayTT, @Phi, @HTTT, @DiaChi <u>Output:</u> Thông tin đơn hàng mới	
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE		SET TRANSACTION ISOLATION LEVEL READ COMMITTED	
BEGIN TRAN			
B1. Xem lần 1 danh sách đơn hàng SELECT d.* FROM DonDatHang d	S(DonDatHang)		

	Xin khóa đọc trên bảng DonDatHang sau đó nhả khóa khi xong		
WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	
		<p>B1: thêm thông tin vào bảng DonDatHang</p> <pre> INSERT dbo.DonDatHang (MaDonHang, MaKhachHang, NgayThanhToan, PhiVanChuyen, HinhThucThanhToan, DiaChiGiaoHang) VALUES (@MaDH, @MaKH, @NgayTT, @PhiVC, @HTTT, @Diachi </pre>	X(DonDatHang)

)	
		COMMIT	
B2. Xem lần 2 danh sách đơn hàng SELECT d.* FROM DonDatHang d	S(DonDatHang)		
COMMIT TRAN			
Thao tác xem thông tin 2 lần thì ở mức LEVEL SERIALIZABLE thì T1 thực thi xong mới chạy T2 nên không xảy ra tình trạng Phantom read			

4. Tình huống 4: Thống kê sản phẩm đã bán của 1 đối tác

ERR01: Phantom read T1 (User = Đối tác): thực hiện xem các đơn hàng chi tiết đã bán và thống kê số lượng sản phẩm đã bán T2 (User = Nhân viên): thực hiện thêm 1 đơn đặt hàng chi tiết mới			
sp_ThongKeSP	Khóa	sp_CTDHMoi	Khóa
<u>Input:</u> @MaDT <u>Output:</u> Danh sách các đơn đặt hàng chi tiết theo sản phẩm và thống kê số lượng		<u>Input:</u> @MaSP, @MaDH, @SL <u>Output:</u> Chi tiết đơn hàng mới	
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE		SET TRANSACTION ISOLATION LEVEL READ COMMITTED	
BEGIN TRAN			
B1: Kiểm tra thông tin @MaDT set tran ISOLATION LEVEL READ COMMITTED	S(DoiTac) //Xin khoá đọc trên bảng Doi Tac.		

<pre>IF not exists (select * from DoiTac where MaDoiTac = @MaDT) begin Print @MaDT + N'Không tồn tại'; rollback tran; end</pre>	<p>sau bước này nhả khóa S</p>		
<p>B2: Xem thông tin sản phẩm của @MaDT</p> <pre>SELECT ct.* FROM DoiTac d, SanPham s, ChiTietDonDatHang ct WHERE d.MaDoiTac=@MaDT and d.MaDoiTac= s.MaDoiTac and ct.MaSanPham= s.MaSanPham</pre>	<p>S(SanPham, DoiTac, CungCapSanPham , ChiNhanh) //Xin khoá đọc trên bảng SanPham, DoiTac, CungCapSanPham, sau đó nhả khóa S</p>		
<p>WAITFOR DELAY '00:00:10'</p>			
		<p>BEGIN TRAN</p>	
		<p>B1: thêm thông tin vào bảng CTDonHang</p> <pre>INSERT dbo.ChiTietDonDatHang (MaDonHang, MaSanPham, Soluong) VALUES</pre>	<p>X(ChiTietDonDatHang)</p>

		(@MaDH, @MaSP, @SL)	
		COMMIT	
B2: thống kê số lượng sản phẩm SELECT SUM(ct.Soluong) as TongSP_Daban FROM DoiTac d, SanPham s, ChiTietDonDatHang ct WHERE d.MaDoiTac=@MaDT and d.MaDoiTac= s.MaDoiTac and ct.MaSanPham= s.MaSanPham	S(SanPham, DoiTac, CungCapSanPham , ChiNhanh) //Xin khoá đọc trên bảng SanPham, DoiTac, CungCapSanPham,		
COMMIT TRAN			
Thao tác xem thông tin 2 lần thì ở mức LEVEL SERIALIZABLE thì T1 thực thi xong mới chạy T2 nên không xảy ra tình trạng Phantom read			

V. Deadlock

1. Tình huống 1: Hai Đối Tác cùng cập nhật giá sản phẩm

ERR05: Conversion deadlock T1 (User = đối tác 1): cập nhật giá sản phẩm trong cửa hàng T2 (User = đối tác 1): cập nhật giá sản phẩm trong cửa hàng			
UpdatePriceProduct_Conversion	Khóa	UpdatePriceProduct_Conversion_2	Khóa
<u>Input:</u> @inputGiaBan float, @inputMaSanPham int <u>Output:</u> giá bán của sản phẩm được cập nhật		<u>Input:</u> @inputGiaBan float, @inputMaSanPham int <u>Output:</u> giá bán của sản phẩm được cập nhật	
SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED		SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED	
BEGIN TRAN			
B1: Kiểm tra tính hợp lệ của mã sản phẩm <pre> if exists (select * from SanPham as SP where SP.MaSanPham = @inputMaSanPham) </pre>	S(SanPham)		

<pre>begin waitfor delay '00:00:08';</pre>			
<pre>B3: Cập nhật giá sản phẩm vào cửa hàng Update SanPham Set GiaBan = @inputGiaBan where MaSanPham = @inputMaSanPham end</pre>	<p>Không cấp được X(GiaBan) vì giao tác kia đã cấp</p>		
<pre>else begin print('San Pham khong ton tai'); end</pre>			

Commit tran		BEGIN TRAN	
		<p>B1: Kiểm tra tính hợp lệ của mã sản phẩm</p> <pre> if exists (select * from SanPham as SP where SP.MaSanPham = @inputMaSanPham) </pre>	S(SanPham)
		begin	
		<p>B3: Cập nhật giá sản phẩm vào cửa hàng</p> <pre> Update SanPham Set GiaBan = @inputGiaBan where MaSanPham = @inputMaSanPham end </pre>	<p>Khóa exclusive lock được cấp cho GiaBan để ghi dữ liệu, khóa được nhả khi hết giao tác</p>

COMMIT		COMMIT	

2. Tình huống 2: Nhân viên cập nhật tình trạng đơn hàng ở bảng DonDatHang và tài xế cập nhật tình trạng đơn hàng ở bảng PhieuGiaoHang

ERR06: Cycle deadlock			
T1 (User = nhân viên 1): cập nhật tình trạng đơn hàng ở bảng DonDatHang			
T2 (User = tài xế 1): cập nhật tình trạng đơn hàng ở bảng PhieuGiaoHang			
NV_CapNhatTTDonHang	Khóa	TX_CapNhatTinhTrangGiaoHang	Khóa
<u>Input:</u> @reqMaDonHang int, @status nvarchar(50) <u>Output:</u> tình trạng đơn hàng của 2 bảng DonDatHang và PhieuGiaoHang được cập nhật		<u>Input:</u> @reqMaPhieuGiao int,@MaDH int, @status nvarchar(50) <u>Output:</u> tình trạng đơn hàng của 2 bảng DonDatHang và PhieuGiaoHang được cập nhật	
<i>SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED</i>		<i>SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED</i>	
BEGIN TRAN			
B1: Kiểm tra tính hợp lệ của mã đơn hàng	S(DonDatHang)		

<pre> if exists (select * from DonDatHang as DDH where DDH.MaDonHang = @reqMaDonHang) </pre>			
B2: Cập nhật tình trạng đơn hàng			
<pre> update DonDatHang set TinhTrangDonHang = @status where MaDonHang = @reqMaDonHang waitfor delay '00:00:07' </pre>	<p>X(TingTrangDonHang) của bảng DonDatHang</p>		
<p>B3: Cập nhật tình trạng đơn hàng ở PHIEUGIAOHANG</p> <pre> Update PhieuGiaoHang set TinhTrangGiaoHang = @status where MaDonHang = @reqMaDonHang end </pre>	<p>X(TingTrangDonHang) không được cấp vì đã cấp khóa bên giao tác 2 trước</p>		

<pre> else begin print('Don hang khong ton tai'); end </pre>			
Commit tran		BEGIN TRAN	
		<p>B1: Kiểm tra tính hợp lệ của mã phiếu giao</p> <pre> if exists (select PGH.MaPhieuGiao from PhieuGiaoHang as PGH where PGH.MaPhieuGiao = @reqMaPhieuGiao and PGH.MaDonHang = @MaDH) </pre>	S(MaPhieuGiao)
		<p>B2: cập nhật tình trạng đơn hàng ở bảng Phiếu Giao hàng</p> <pre> begin </pre>	X(TingTrangDonHang) của bảng

		<pre> Update PhieuGiaoHang set TinhTrangGiaoHang = @status where MaPhieuGiao = @reqMaPhieuGiao </pre>	PhieuGiaoHang
		<p>B3: Cập nhật tình trạng đơn hàng ở Đơn đặt hàng</p> <pre> Update DonDatHang set TinhTrangDonHang = @status where MaDonHang = @MaDH end </pre>	X(TingTrangDonHang) của bảng DonDatHang
		<pre> else begin print('Don hang khong ton tai') end </pre>	
COMMIT		COMMIT	

1.