BÁO CÁO ĐỒ ÁN THỰC HÀNH

MÔN HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU

GVHD: Tiết Gia Hồng

MỤC LỤC

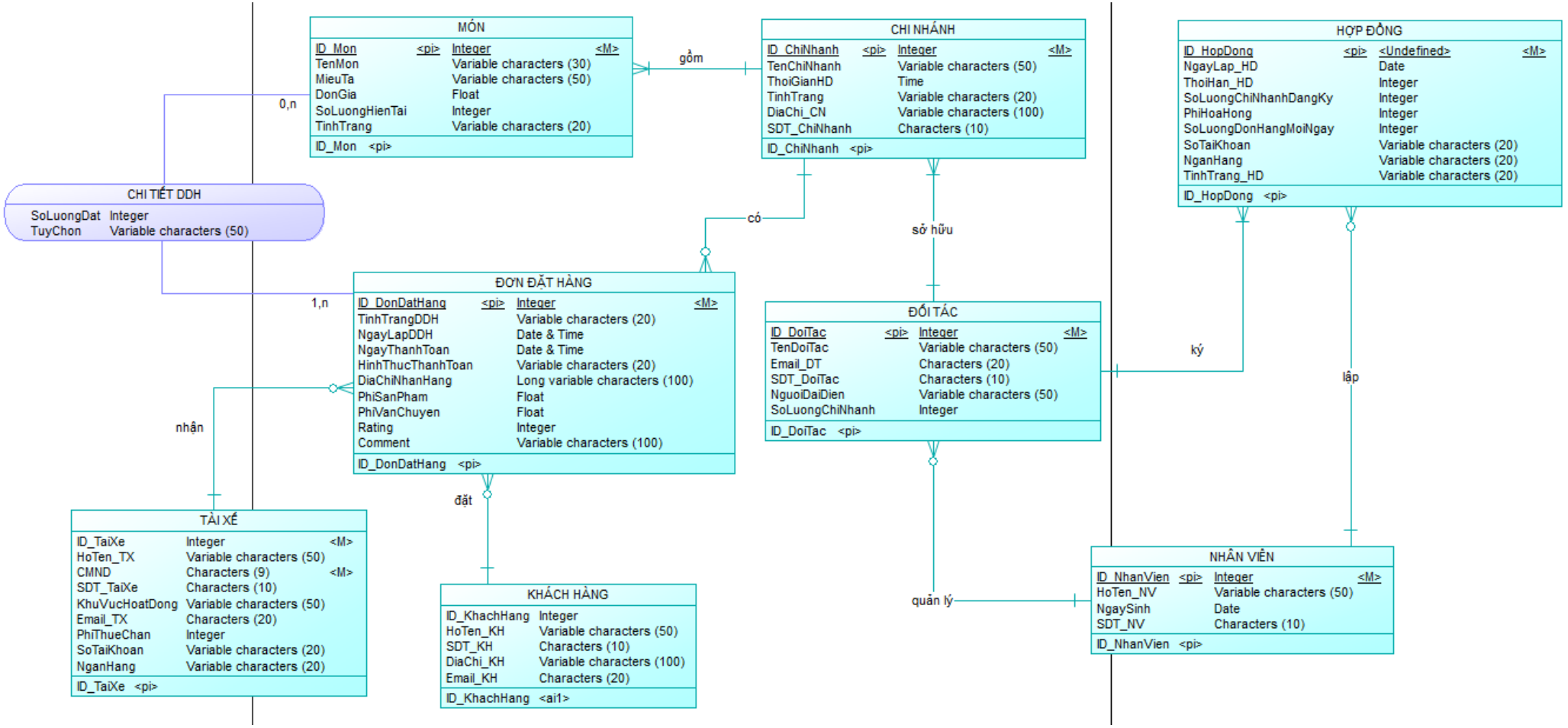
THÔNG TIN NHÓM 3
BẢNG THIẾT KẾ THỰC THỂ KẾT HỢP 4
I. Lược đồ thực thể kết hợp 4
II. Ràng buộc dữ liệu bổ sung 5
III. Mô hình cơ sở dữ liệu quan hệ 9
IV. LOGIN, USER, ROLE và PERMISSION 10
CÀI ĐẶT TÌNH HUỐNG TRANH CHẤP 12
I. Sinh viên thực hiện: Mai Nhật Nam 12
II. Sinh viên thực hiện: Nguyễn Đặng Nam Khánh 21
III. Sinh viên thực hiện: Minh Triết 29
IV. Sinh viên thực hiện: Lâm Quang Duy 38
GIẢI QUYẾT TÌNH HUỐNG TRANH CHẤP 51
I. Sinh viên thực hiện: Mai Nhật Nam 51
II. Sinh viên thực hiện: Nguyễn Đặng Nam Khánh 64
III. Sinh viên thực hiện: Minh Triết 73
IV. Sinh viên thực hiện: Lâm Quang Duy 81

THÔNG TIN NHÓM

STT	MSSV	Họ tên	Công việc	% Hoàn thành
1	20120139	Mai Nhật Nam	ER, mô hình quan hệ, xác định phân quyền, tranh chấp, xử lý tranh chấp.	90%
2	20120115	Nguyễn Đặng Nam Khánh	ER, xác định phân quyền, tranh chấp, xử lý tranh chấp.	90%
3	20120223	Thái Minh Triết	ER, mô hình quan hệ, tranh chấp, xử lý tranh chấp.	90%
4	20120065	Lâm Quang Duy	Script phân quyền, tranh chấp, xử lý tranh chấp.	80%

BẢNG THIẾT KẾ THỰC THỂ KẾT HỢP

I. Lược đồ thực thể kết hợp



II. Ràng buộc dữ liệu bổ sung

1. Số lượng chi nhánh đăng ký của hợp đồng không được lớn hơn số lượng chi nhánh của đối tác

- Loại RBTV: Liên thuộc tính liên quan hệ
- Bối cảnh: Đối tác, hợp đồng
- Nội dụng: (∀t) (Đối tác(t) ∧ (∀s) (Hợp đồng(s) ∧ (t.ID = s.ID) ^ t.SoLuongChiNhanh >= s.SoLuongChiNhanhDangKy)))
- Bảng tầm ảnh hưởng:

	T	X	S
Đối tác	-	-	+ (SoLuongChiNhanh)
Hợp đồng	+	-	+ (SoLuongChiNhanhDangKy, idDoiTac)

2. Số CMND/CCCD của tài xế không được trùng

- Loại RBTV: Liên bộ
- Bối cảnh: Tài xế
- Nội dung: (∀t1, t2) (TaiXe(t1) ∧ TaiXe(t2) ∧ (t1 ≠t2 => t1.CMND ≠ t2.CMND))
- Bảng tầm ảnh hưởng:

	T	X	S
TaiXe	+	-	+ (CMND)

3. Tình trạng hợp đồng phải là 'Còn hiệu lực' hay 'Hết hiệu lực'

Loại RBTV: Miền giá trị

Bối cảnh: Hợp đồng

Nội dung: (∀t) HopDong(t) ∧ (t.TinhTrang=N' Còn hiệu lực' or t.TinhTrang = N'Hết hiệu lực')

Bảng tầm ảnh hưởng:

	T	X	S
Hợp đồng	+	-	+(TinhTrang)

4. Tình trạng chi nhánh chỉ có thể là Bình thường hoặc Tạm nghỉ

- Loại RBTV: Miền giá trị
- Bối cảnh: Chi nhánh
- Nội dung: (∀t) ChiNhanh(t) ∧ (t.TinhTrang = N'Bình thường' or t.TinhTrang = N'Tạm nghỉ')
- Bảng tầm ảnh hưởng:

	T	X	S
Chi nhánh	+	-	+(TinhTrangChiNhanh)

5. Mỗi chi nhánh phải có ít nhất một món thuộc về chi nhánh đó

- Loại RBTV: Liên bộ liên quan hệ
- Bối cảnh: Chi nhánh, Món
- Nội dung: (∀t) (ChiNhanh(t) ∧ (∃s) (Mon(s) ∧ (t.ID = s.idChiNhanh)))
- Bảng tầm ảnh hưởng:

	T	X	S
Chi nhánh	+	-	-
Món	-	+	+ (idChiNhanh)

6. Một đơn đặt hàng phải thuộc một khách hàng

- Loại RBTV: tham chiếu
- Bối cảnh: Đơn đặt hàng, khách hàng
- Nội dung: (∀t) DonDatHang(t) ^ (∃s) KhachHang(s) ^ t.idKhachHang = s.ID

- Bảng tầm ảnh hưởng:

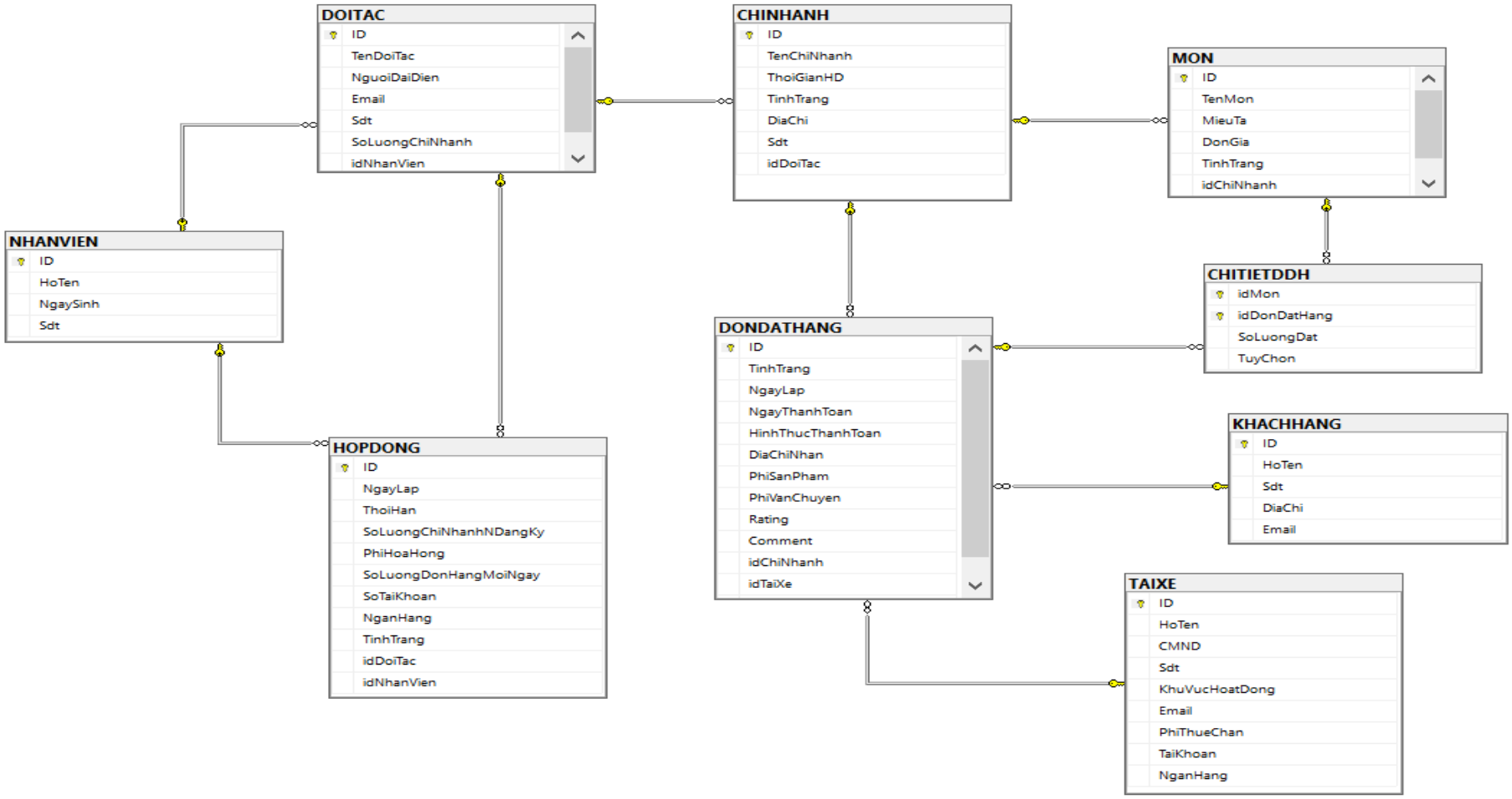
	T	X	S
Đơn đặt hàng	+	-	+ (idKhachHang)
Khách hàng	-	+	-

7. Tài xế chỉ có thể giao những đơn đặt hàng có địa chỉ giao trong khu vực hoạt động mà tài xế đã đăng ký

- Loại RBTV: Liên thuộc tính liên quan hệ
- Bối cảnh: Đơn đặt hàng, Tài xế
- Nội dung: (∀t) DonDatHang(t) ∧ (∀s) (TaiXe(s) ∧ (t.idTaiXe = s.ID and t.DiaChiGiaoHang LIKE '%s.KhuVucHoatDong%')))
- Bảng tầm ảnh hưởng:

	T	X	S
Đơn đặt hàng	-	+	+ (DiaChiGiaoHang, idTaiXe)
Tài xế	-	+	+ (KhuVucHoatDong)

III. Mô hình cơ sở dữ liệu quan hệ



IV. LOGIN, USER, ROLE và PERMISSION

1. **ADMIN:**
   Có quyền xem, thêm, xóa, sửa tất cả các bảng trong cơ sơ dữ liệu.
2. **Nhân viên**

Bảng/Quyền	Xem	Thêm	Xóa	Sửa
Hợp đồng	x	x	x	x
Đối tác	x	x	x	
Chi nhánh	x	x	x	

3. **Đối tác:**

Bảng/Quyền	Xem	Thêm	Xóa	Sửa
Chi nhánh	x			x
Hợp đồng	x			

4. **Chi nhánh:**

Bảng/Quyền	Xem	Thêm	Xóa	Sửa
Món	x	x	x	x
Đơn đặt hàng	x			x
Chi tiết DDH	x			

5. **Khách hàng:**

Bảng/Quyền	Xem	Thêm	Xóa	Sửa
Đơn đặt hàng	x	x	x	x
Chi tiết DDH	x			x
Đối tác	x			
Món	x			
Chi nhánh	x			

6. **Tài xế:**

Bảng/Quyền	Xem	Thêm	Xóa	Sửa
Đơn đặt hàng	x			x

CÀI ĐẶT TÌNH HUỐNG TRANH CHẤP

I. Sinh viên thực hiện: Mai Nhật Nam

1. Tình huống 1: Tài xế nổ đơn thấy tình trạng đơn hàng đang là 'Chờ nhận', cùng lúc đó khách hàng muốn xóa đơn hàng của hàng của mình mà tình trạng là 'Chờ nhận' nên khách hàng click Xóa đơn đặt hàng. Điều này khiến cho tài xế không thể cập nhật để giao cho đơn đặt hàng này.

ERR01: ***Unrepeatable Read giải quyết không cẩn thận bị Conversion Lock***
T1 (User = tài xế): thực hiện cập nhật thông tin đơn hàng A để giao hàng.
T2 (User = khách hàng): thực hiện xóa đơn hàng A của mình (kèm điều kiện tình trạng đơn hàng "Chờ nhận").

SP_TAIXE_UPDATE_DONHANG	Khóa	SP_KHACHHANG_DELETE_DONHANG	Khóa
Input: *Mã tài xế, mã đơn đặt hàng.* ***Output***: cập nhật đơn đặt hàng.		***Input***: *Mã khách hàng, mã đơn hàng.* ***Output***: Xóa đơn đặt hàng trên.	
SET TRANSACTION ISOLATION LEVEL ……		SET TRANSACTION ISOLATION LEVEL ……	
BEGIN TRAN			
B1: Kiểm tra tài xế có tồn tại IF NOT EXISTS (select * from TAI_XE where MaTX = 2) begin print 'Tài xế không tồn tại' rollback tran return end			
B2: Kiểm tra thông tin đơn đặt hàng IF EXISTS (select * from DON_DAT_HANG where MaDDH = 1) Begin Select * from DON_DAT_HANG where MaDDH = 1			

WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	
		B1: Kiểm tra khách hàng có tồn tại IF NOT EXISTS (select * from KHACH_HANG where KhachHang = 1) Begin Print 'Khách hàng không tồn tại' Rollback tran Return End	

		B2: Khách hàng DELETE đơn hàng
		
		IF (EXISTS (select * from DON_DAT_HANG where
		MaDDH = DH001 AND TinhTrang = 'Chờ nhận'))
		Begin
		DELETE DON_DAT_HANG
		Where MaDDH = 1 AND MaKH = 1
		End	

		COMMIT TRAN	
B3: Kiểm tra tình trạng đơn hàng UPDATE DON_DAT_HANG set TinhTrang = 'Đã nhận' , MaTX = 2 where MaDDH = 1 end			
COMMIIT TRAN			

2. Tình huống 2: Tài xế nổ đơn hàng A nên chuẩn bị cập nhật để giao đơn hàng A, sau đó khách hàng muốn xóa đơn hàng A của mình nên đã vào cập nhật tình trạng đơn hàng A từ 'Chờ nhận' sang 'Đã hủy'. Sau đó tài xế cập nhật tình trạng 'Đã nhận đơn hàng' ghi đè lên kết quả cập nhật của khách hàng.

ERR02: *Lost Update*
T1 (User = Khách hàng): thực hiện cập nhật thông tin đơn đặt hàng.
T2 (User = Tài xế): thực hiện cập nhật thông tin đơn đặt hàng.

SP_TAIXE_UPDATE_DONHANG	Khóa	sp_KhachHangCapNhatDonHang	Khóa
Input**: Mã tài xế, mã đơn đặt hàng.* ***Output: Cập nhật đơn đặt hàng.		***Input***: *Mã khách hàng, mã đơn đặt hàng.* ***Output***: Cập nhật đơn đặt hàng.	
SET TRANSACTION ISOLATION LEVEL ……..		SET TRANSACTION ISOLATION LEVEL ……..	
BEGIN TRAN			
B1: Kiểm tra thông tin tài xế IF NOT EXISTS (select * from TAI_XE where MaTX = TX001) Begin Print 'Tài xế không tồn tại' Rollback tran Return End			
B2: Kiểm tra thông tin đơn đặt hàng IF EXISTS (select * from DON_DAT_HANG where MaDDH = DH001) Begin Select * from DON_DAT_HANG where MaDDH = DH001 End			
WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	
		B1: Kiểm tra khách hàng có tồn tại IF NOT EXISTS (select * from KHACH_HANG where KhachHang = KH002)	

		Begin Print 'Khách hàng không tồn tại' Rollback tran Return End	

		B2: Khách hàng UPDATE tình trạng
		
		IF (EXISTS (select * from DON_DAT_HANG where MaDDH = DH001 AND TinhTrang = 'Chờ nhận'))
		Begin
		UPDATE DON_DAT_HANG
		Set TinhTrang = 'Đã hủy'
		Where MaDDH = DH001
		End	

		COMMIT TRAN	

B3: Kiểm tra tình trạng đơn hàng
IF (select TinhTrang from DON_DAT_HANG where MaDDH = DH001)
Begin			

UPDATE DON_DAT_HANG Set TinhTrang = 'Đã nhận', TaiXe = 2 Where MaDDH = DH001 End			
COMMIT TRAN			

3. Tình huống 3: Cuối tháng đối tác vào thống kê xem tháng này trên hệ thống online mình bán được bao nhiêu bao nhiêu đơn hàng. Ngay sau đó khách hàng vào đặt thêm một vài đơn đặt hàng. Và sau đó, đối tác muốn in ra xem danh sách đơn đặt hàng thì thấy con số thống kê ở trên và số dòng mình in ra không khớp. Tại sao?

ERR03: Phantom
T1 (User = Đối tác): thực hiện thống kê số lượng đơn hàng và xem thông tin đơn đặt hàng.
T2 (User = Khách hàng): thực hiện thêm đơn đặt hàng.

sp_DoiTacThongKe	Khóa	sp_KhachHangInsertDDH	Khóa
Input**: Mã đối tác.* ***Output: Số lượng đơn hàng, danh sách đơn hàng.		***Input***: ***Mã khách hàng, mã đơn đặt hàng, địa chỉ nhận, phí vận chuyển, hình thức thanh toán, mã chi nhánh, số lượng sản phẩm đặt, mã món.*** ***Output***: Thêm đơn đặt hàng.	
SET TRANSACTION ISOLATION LEVEL ……		SET TRANSACTION ISOLATION LEVEL ……	
BEGIN TRAN			
B1: Kiểm tra thông tin đối tác IF NOT EXISTS (select * from DOI_TAC where MaDT = DT001) Begin Print 'Đối tác không tồn tại' Rollback tran Return End			
B2: Thống kê số lượng đơn đặt hàng Declare @SL_DonHang int Set @SL_DonHang = (select COUNT (*) From DOI_TAC DT, CHI_NHANH CN, DON_DAT_HANG DH Where DT. MaDT = DT001 and DT. MaDT=CN.MaDT and CN.MaCN = DH.MaCN)			

B3: In số lượng đơn đặt hàng Print 'Số lượng đơn hàng của đối tác' + DH001 + 'là' @SL_DonHang			
WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	
		B1: Kiểm tra thông tin của khách hàng IF NOT EXISTS (select * from KHACH_HANG where KhachHang= KH002) Begin Print 'Khách hàng không tông tại' Rollback tran Return End	
		B2: Kiểm tra mã đơn đặt hàng IF EXISTS (select * from DON_DAT_HANG where MaDDH = DH001) Begin PRINT 'Mã đơn hàng đã tồn tại' Set TinhTrang = 'Đã hủy' Where MaDDH = DH001 End	
		B3: Xử lý thông tin Decalre @DonGia float Declare @PhiSanPham Declare @ThanhTien Set @DonGia = (select Gia from MON where MaMon = @MaMon)	

		Set @PhiSanPham = @DonGia * SL_SanPham	
		B4: Thêm đơn đặt hàng INSERT INTO DON_DAT_HANG VALUES (@MaDDH, N'Chờ nhận', @DiaChiNhan, @PhiSanPham, @PhiVanChuyen, @HT_ThanhToan, GETDATE (), NULL, @MaKH, @MaChiNhanh) INSERT INTO CHI_TIET_DDH VALUES (@MaMon, @MaDDH, @SL_SanPham)	
		COMMIT TRAN	
B4: In danh sách đơn đặt hàng Select * From DOITAC DT, CHI_NHANH CN, DON_DAT_HANG DDH Where DT. MaDoiTac = @MaDT and DT. MaDoiTac = CN. DoiTac and CN. MaChiNhanh= DDH.ChiNhanh			
COMMIT TRAN			

4. Tình huống 4: Khi đến thời gian gia hạn hợp đồng. Tại một thời điểm 2 nhân viên vào đọc số lượng chi nhánh đăng ký trên hợp đồng. Một người thì xóa còn một người thì thêm đồng thời cập nhật thuộc tính số lượng chi nhánh khiến cho nó không đồng nhất.

ERR04: Cycle Lock T1 (User = Nhân viên): thực hiện thêm chi nhánh và cập nhật ở hợp đồng. T2 (User = Khách hàng): thực hiện xóa chi nhánh và cập nhật ở hợp đồng.			
SP_NhanVienThemChiNhanh	**Khóa**	**sp_NhanVienXoaChiNhanh**	**Khóa**

Input**: Mã đối tác, tên chi nhánh, thời gian hoạt động, tình trạng, địa chỉ, số điện thoại.* ***Output: Thêm 1 chi nhánh cho đối tác và cập nhật số lượng chi nhánh bên hợp đồng của đối tác.		***Input***: *Mã đối tác, tên chi nhánh.* ***Output***: Xóa 1 chi nhánh cho đối tác	
SET TRANSACTION ISOLATION LEVEL ……		SET TRANSACTION ISOLATION LEVEL ……	
BEGIN TRAN			
B1: Kiểm tra thông tin đối tác IF NOT EXISTS (select * from DOI_TAC where MaDT = 1) Begin Print 'Đối tác không tồn tại' Rollback tran Return End			
B2: Xem số lượng chi nhánh đăng ký hiện tại của đối tác. Declare @SLChiNhanh int Set @SL_DonHang = (select soluongChiNhanh From HOP_DONG Where DT. MaDT = 1)			
B3: In số lượng đơn đặt hàng Print 'Số lượng chi nhánh của đối tác trước khi thêm là' + @SLChiNhanh	**Ví dụ: In ra 3**		
WAITFOR DELAY '00:00:3'			
		BEGIN TRAN	

		B1: Kiểm tra thông tin đối tác IF NOT EXISTS (select * from DOI_TAC where MaDT = 1) Begin Print 'Đối tác không tồn tại' Rollback tran Return End	
		B2: Xem số lượng chi nhánh đăng ký hiện tại của đối tác. Declare @SLChiNhanh int Set @SL_DonHang = (select soluongChiNhanh From HOP_DONG Where DT. MaDT = 1)	
		B3: In số lượng đơn đặt hàng Print 'Số lượng chi nhánh của đối tác trước khi thêm là' + @SLChiNhanh	**Ví dụ: In ra 3**
		WAITFOR DELAY '00:00:3'	
B4: Thêm chi nhánh cho đối tác Insert into ChiNhanh values (TenChiNhanh, ThoiGianHD, TinhTrang, DiaChi, Sdt, idDoiTac)			
WAITFOR DELAY '00:00:3'			

		B4: Cập nhật số lượng chi nhánh đăng ký Update HOP_DONG set SoLuongCNDangKy = @SLChiNhanh – 1 where idDoiTac = 1	soLuong = 2
		WAITFOR DELAY '00:00:3'	
B5: Cập nhật số lượng chi nhánh đăng ký Update HOP_DONG set SoLuongCNDangKy = @SLChiNhanh + 1 where idDoiTac = 1	soLuong = 4		
COMMIT TRAN			
		B5: Xóa chi nhánh Delete ChiNhanh where TenChiNhanh = 'Chi nhánh 4' and idDoiTac = 1	
		COMMIT TRAN	

II. Sinh viên thực hiện: Nguyễn Đặng Nam Khánh

5. Tình huống 1: Tài xế đầu tiên thấy tình trạng đơn hàng là chờ nhận nên chọn đơn hàng để giao, cùng lúc đó tài xế khác cũng thấy tình trạng đơn hàng là chờ nhận nên cũng chọn để giao. Hệ thống sẽ thông báo cả cho cả 2 là đều nhận đơn hàng thành công nhưng thực ra chỉ có tài xế 1 là nhận được đơn hàng.

ERROR 1: Lost Update T1(User=Tài xế): thực hiện chỉnh sửa đơn đặt hàng T2(User=Tài xế): thực hiện chỉnh sửa đơn đặt hàng			
USP_SUADDH_TAIXE1	**Khóa**	**USP_SUADDH_TAIXE2**	**Khóa**
Input*: Mã tài xế, mã đơn đặt hàng** ***Output: Tài xế nhận được đơn đặt hàng		***Input***: **Mã tài xế, mã đơn đặt hàng** ***Output***: Tài xế nhận được đơn đặt hàng	
SET TRANSACTION ISOLATION LEVEL ……		SET TRANSACTION ISOLATION LEVEL ……	
BEGIN TRAN			

B1: Kiểm tra tài xế có tồn tại
 	
IF (NOT EXISTS(select* from TaiXe where idTaiXe=@MaTaiXe))
BEGIN
print N'Tài xế không tồn tại'
rollback tran
return
END			

B2: Kiểm tra đơn đặt hàng
IF (EXISTS (select * from DonDatHang where idDonDatHang=@MaDDH and idTaiXe IS NULL and TinhTrang=N'Chờ nhận'))
BEGIN			

WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	

		B1: Kiểm tra tài xế có tồn tại
		
		IF (NOT EXISTS(select* from TaiXe where idTaiXe=@MaTaiXe))
		BEGIN
		print N'Tài xế không tồn tại'
		rollback tran
		return
		END	

		B2: Kiểm tra đơn đặt hàng IF (EXISTS (select * from DonDatHang where idDonDatHang=@MaDDH and idTaiXe IS NULL and TinhTrang=N'Chờ nhận'))	
B3: Cập nhật đơn đặt hàng			

UPDATE DonDatHang SET TinhTrang=N'Đã nhận', idTaiXe=@MaTaiXe where idDonDatHang=@MaDDH print N'Nhận đơn hàng thành công' END end			
		B3: Cập nhật đơn đặt hàng UPDATE DonDatHang SET TinhTrang=N'Đã nhận', idTaiXe=@MaTaiXe where idDonDatHang=@MaDDH print N'Nhận đơn hàng thành công' END	
COMMIIT TRAN			
		COMMIT TRAN	

6. Tình huống 2: Chi nhánh thứ nhất sửa món ăn, cùng lúc đó chi nhánh thứ 2 thực hiện xóa món ăn đó làm thao tác sửa món ăn của chi nhánh đầu tiên không thành công.

ERROR 2: Unrepeatable read **T1(User=Chi nhánh): thực hiện chỉnh sửa món** **T2(User=Chi nhánh): thực hiện xóa món**			
USP_SUAMON_CHINHANH	**Khóa**	USP_XOAMON_CHINHANH	**Khóa**
Input **Mã món, tên món mới, giá món mới, tình trạng** ***Output***: Món đã thay đổi		***Input***: **Mã món, mã chi nhánh** ***Output***: Xóa món	

SET TRANSACTION ISOLATION LEVEL ……		SET TRANSACTION ISOLATION LEVEL ……	
BEGIN TRAN			

B1: Kiểm tra chi nhánh có tồn tại hay không
IF (NOT EXISTS (select* from ChiNhanh where idChiNhanh=@MaCN))
BEGIN
print N'Chi nhánh không tồn tại'
rollback tran
return
END			

B2: Kiểm tra món có tồn tại hay không
IF (EXISTS (select * from Mon where idMon=@MaMon and idChiNhanh=@MaCN))
BEGIN			

WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	

		B1: Kiểm tra chi nhánh có tồn tại hay không
		IF (NOT EXISTS (select* from ChiNhanh where idChiNhanh=@MaCN))
		BEGIN
		print N'Chi nhánh không tồn tại'
		rollback tran
		return
		END	

		B2: Xóa món
		IF (EXISTS (select * from Mon where idMon=@MaMon and idChiNhanh=@MaCN))
		BEGIN
		DELETE from Mon where idMon=@MaMon
		print N'Xóa món thành công'
		select * from Mon
		END	

B3: Cập nhật thông tin cho món UPDATE Mon SET TenMon=@TenMon,DonGia=@Gia,TinhTrang=@TinhTrang where idMon=@MaMon print N'Sửa món thành công' select * from Mon where idMon=@MaMon END			
		COMMIT TRAN	
COMMIT TRAN			

7. Tình huống 3: Khách hàng muốn xem chi tiết một món ăn nhưng cùng lúc đó chi nhánh xóa món ăn đó làm khách hàng không thể xem được chi tiết món ăn

ERROR 3: Unrepeatable read **T1(User=Khách hàng): thực hiện xem món** **T2(User=Chi nhánh): thực hiện xóa món**			
USP_XEMMON_KHACHHANG	**Khóa**	USP_XOAMON_CHINHANH	**Khóa**

Input: **Mã món, mã chi nhánh** ***Output***: Chi tiết món ăn		***Input***: **Mã món, mã chi nhánh** ***Output***: Xóa món
SET TRANSACTION ISOLATION LEVEL ……		SET TRANSACTION ISOLATION LEVEL ……
BEGIN TRAN		

B1: Kiểm tra chi nhánh có tồn tại hay không
IF (NOT EXISTS (select* from ChiNhanh where idChiNhanh=@MaCN))
BEGIN
print N'Chi nhánh không tồn tại'
rollback tran
return
END		

B2: Kiểm tra món có tồn tại hay không
IF (EXISTS (select * from Mon where idMon=@MaMon and idChiNhanh=@MaCN))
BEGIN		

WAITFOR DELAY '00:00:10'		
		BEGIN TRAN

		B1: Kiểm tra chi nhánh có tồn tại hay không
		IF (NOT EXISTS (select* from ChiNhanh where idChiNhanh=@MaCN))
		BEGIN
		print N'Chi nhánh không tồn tại'
		rollback tran
		return
		END

		B2: Xóa món

		IF (EXISTS (select * from Mon where idMon=@MaMon and idChiNhanh=@MaCN))
		BEGIN
		DELETE from Mon where idMon=@MaMon
		print N'Xóa món thành công'
		select * from Mon
		END	

B3: Xuất thông tin món declare @ten nvarchar(50),@gia real,@tinhtrang nvarchar(50) select @ten=TenMon,@gia=DonGia,@tinhtrang=TinhTrang from Mon where idMon=@MaMon print N'Tên món: '+@ten print N'Giá: '+@gia print N'Tình trạng: '+@tinhtrang			
		COMMIT TRAN	
COMMIT TRAN			

8. Tình huống 4: Tài xế cập nhật dữ liệu cho đơn hàng nhưng dữ liệu đơn hàng đưa vào không hợp lệ nên hệ thống rollback giao tác trên, cùng lúc đó thì khách hàng xem dữ liệu đơn hàng sẽ bị xem sai dữ liệu.

ERROR 4: Dirty Read **T1(User=Tài xế): thực hiện sửa đơn đặt hàng** **T2(User=Khách hàng): thực hiện xem món**			
USP_SUADDH_TAIXE	**Khóa**	USP_XEMMON_KHACHHANG	**Khóa**
Input*: Mã đơn đặt hàng, mã tài xế, tình trạng đơn đặt hàng** ***Output: Cập nhật dữ liệu cho đơn đặt hàng		***Input***: **Mã đơn đặt hàng, mã khách hàng** ***Output***: Chi tiết đơn đặt hàng	

SET TRANSACTION ISOLATION LEVEL ……..		SET TRANSACTION ISOLATION LEVEL ……..	
BEGIN TRAN			

B1: Kiểm tra tài xế có tồn tại hay không
IF (NOT EXISTS(select* from TaiXe where idTaiXe=@MaTaiXe))
BEGIN
print N'Tài xế không tồn tại'
rollback tran
return
END			

B2: Cập nhật tình trạng đơn đặt hàng UPDATE DonDatHang SET TinhTrang=@TinhTrang, idTaiXe=@MaTaiXe where idDonDatHang=@MaDDH print N'Nhận đơn hàng thành công' select * from DonDatHang where idDonDatHang=@MaDDH			
WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	

		IF (NOT EXISTS(select* from KhachHang where idKhachHang=@MaKH))
		BEGIN
		print N'Khách hàng không tồn tại'
		rollback tran
		return	

		END	
		B2: Xuất thông tin đơn đặt hàng declare @ma nvarchar(50),@gia real,@tinhtrang nvarchar(20) select @ma=idDonDatHang,@gia=PhiSanPham+PhiVanChuyen,@tinhtrang=TinhTrang from DonDatHang where idDonDatHang=@MaDDH print N'Mã đơn đặt hàng: '+@ma print N'Giá: '+ CAST (@gia as varchar(20)) print N'Tình trạng:' +@tinhtrang	
IF EXISTS (select * from DonDatHang where TinhTrang NOT IN (N'Đã nhận',N'Đang giao',N'Chờ duyệt',N'Đang chờ')) BEGIN rollback tran END			
		COMMIT TRAN	
COMMIT TRAN			

III. Sinh viên thực hiện: Minh Triết

9. Tình huống 1: Chi nhánh 1 thay đổi thông tin món ăn, cùng lúc chi nhánh 2 cũng thay đổi thông tin của món ăn đó dẫn đến lost update.

ERR01: Lost Update **T1 (User = Chi Nhánh): thực hiện cập nhật thông tin món ăn** **T2 (User = Chi Nhánh): thực hiện cập nhật thông tin món ăn**			
sp_SuaMonAn_ChiNhanh1	**Khóa**	**sp_SuaMonAn_ChiNhanh2**	**Khóa**
***Input*: Mã Món ăn, Mã chi nhánh**		***Input*: Mã món ăn, Mã chi nhánh**	

Output: *cập nhật món ăn*		***Output***: *Cập nhật thông tin món ăn*	
SET TRANSACTION ISOLATION LEVEL ……..		SET TRANSACTION ISOLATION LEVEL ……..	
BEGIN TRAN			
B1: Kiểm tra món ăn có tồn tại if not exists (select* from MON where idMon = @MaMon and idChiNhanh = @MaChiNhanh) begin print N'Món ăn này không tồn tại!' rollback tran return end			
B2: Nếu món ăn có tồn tại if exists (select * from MON where idMon = @MaMon and idChiNhanh = @MaChiNhanh) begin			
WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	
		B1: Kiểm tra món ăn có tồn tại if not exists (select* from MON where idMon = @MaMon and idChiNhanh = @MaChiNhanh) begin print N'Món ăn này không tồn tại!' rollback tran return end	
		B2: Nếu món ăn có tòn tại	

		if exists (select * from MON where idMon = @MaMon and idChiNhanh = @MaChiNhanh) begin	
B3: Cập nhật lại món ăn update MON set MieuTa = @MieuTa where idMon = @MaMon and TenMon = @TenMon and idChiNhanh = @MaChiNhanh update MON set DonGia = @Gia where idMon = @MaMon and TenMon = @TenMon and idChiNhanh = @MaChiNhanh print N'Sửa món ăn thành công' select * from MON where idMon = @MaMon and idChiNhanh = @MaChiNhanh end			
		B3: Cập nhật lại món ăn update MON set MieuTa = @MieuTa where idMon = @MaMon and TenMon = @TenMon and idChiNhanh = @MaChiNhanh update MON set DonGia = @Gia where idMon = @MaMon and TenMon = @TenMon and idChiNhanh = @MaChiNhanh print N'Sửa món ăn thành công'	

		select * from MON where idMon = @MaMon and idChiNhanh = @MaChiNhanh end	
COMMIIT TRAN			
		COMMIIT TRAN	

10. Tình huống 2: Nhân viên 1 thay đổi ngày kết thúc của hợp đồng cùng lúc đó nhân viên 2 cũng vào thay đổi ngày kết thúc của hợp đồng đó dẫn đến Lost update

ERR02: Lost Update **T1 (User = Nhân viên): thực hiện thay đổi ngày kết thúc của hợp đồng** **T2 (User = Nhân viên): thực hiện thay đổi ngày kết thúc của hợp đồng**			
sp_ChinhSuaHD_NhanVien1	**Khóa**	**sp_ChinhSuaHD_NhanVien2**	**Khóa**
Input*: Mã hợp đồng, mã nhân viên, mã đối tác** ***Output: Cập nhật hợp đồng		***Input***: ***Mã hợp đồng, mã nhân viên, mã đối tác.*** ***Output***: Cập nhật hợp đồng	
SET TRANSACTION ISOLATION LEVEL ……		SET TRANSACTION ISOLATION LEVEL ……	
BEGIN TRAN			
B1: Kiểm tra thông tin hợp đồng if not exists (select * from HOPDONG where idHopDong = @MaHD and idNhanVien = @MaNv and idDoiTac = @MaDoiTac) begin print N'Hợp đồng này không tồn tại' rollback tran return end			

B2: Nếu hợp đồng có tồn tại if exists (select * from HOPDONG where idHopDong = @MaHD and idNhanVien = @MaNv and idDoiTac = @MaDoiTac) begin			
WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	
		B1: Kiểm tra hợp đồng if not exists (select * from HOPDONG where idHopDong = @MaHD and idNhanVien = @MaNv and idDoiTac = @MaDoiTac) begin print N'Hợp đồng này không tồn tại' rollback tran return end	
		B2: Nếu hợp đồng tồn tại if exists (select * from HOPDONG where idHopDong = @MaHD and idNhanVien = @MaNv and idDoiTac = @MaDoiTac) begin	
B3: Tiến hành cập nhật lại hợp đồng update HOPDONG set ThoiHan = @ThoiHan where idHopDong = @MaHD and idNhanVien = @MaNv and idDoiTac = @MaDoiTac end			

		B3: Tiến hành cập nhật lại hợp đồng update HOPDONG set ThoiHan = @ThoiHan where idHopDong = @MaHD and idNhanVien = @MaNv and idDoiTac = @MaDoiTac end	
COMMIT TRAN			
		COMMIT TRAN	

11. Tình huống 3: Khánh hàng 1 vào xem và đặt món ăn, cùng lúc đó khách hàng 2 cũng vào xem và đặt món ăn đó làm cho cho việc xem món ăn của khách hàng 1 bị sai gây ra LOST UPDATE

ERR03: Lost Update **T1 (User = Khách Hàng 1): thực hiện đặt món ăn** **T2 (User = Khách Hàng 2): thực hiện đặt món ăn**			
sp_DatMonAn_KhachHang1	**Khóa**	**sp_DatMonAn_KhachHang2**	**Khóa**
Input**: Mã món ăn, Mã Chi Nhánh, Mã Khách Hàng, Số lượng đặt của món đó* ***Output: Thêm đơn đặt hàng		***Input***: *Mã món ăn, Mã Chi Nhánh, Mã Khách Hàng, Số lượng đặt của món đó* ***Output***: Thêm đơn đặt hàng.	
SET TRANSACTION ISOLATION LEVEL ……..		SET TRANSACTION ISOLATION LEVEL ……..	
BEGIN TRAN			
B1: Kiểm tra thông tin món ăn if not exists (select* from MON where idMon = @MaMon and idChiNhanh = @MaChiNhanh) begin print N'Món ăn này không tồn tại!'			

rollback tran return end			
B2: Nếu món ăn tồn tại if exists (select* from MON where idMon = @MaMon and idChiNhanh = @MaChiNhanh) begin			

B3: Kiểm tra số lượng món ăn
begin
declare @Gia real, @SLHT int
select @Gia = DonGia, @SLHT = SoLuongHienTai
from MON where idMon = @MaMon
if (@SLHT < @SoLuongMua)
print N'Món bạn đặt đã hết ăn cái khác đi <3'			

WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	
		B1: Kiểm tra thông tin món ăn if not exists (select* from MON where idMon = @MaMon and idChiNhanh = @MaChiNhanh) begin print N'Món ăn này không tồn tại!' rollback tran return end	
		B2: Nếu món ăn tồn tại	

		if exists (select* from MON where idMon = @MaMon and idChiNhanh = @MaChiNhanh) begin	

		B3: Kiểm tra số lượng món ăn
		begin
		declare @Gia real, @SLHT int
		select @Gia = DonGia, @SLHT = SoLuongHienTai from MON where idMon = @MaMon
		if (@SLHT < @SoLuongMua)
		print N'Món bạn đặt đã hết ăn cái khác đi <3'	

B4: Thêm món ăn vào đơn đặt hàng insert into DonDatHang VALUES(N'Chờ nhận','2022-12-01',NULL,'Online',N'121/8 Phạm Văn Đồng, Quận Bình Thạnh',@Gia*@SoLuongMua,3000,NULL,NULL,@MaChiNhanh,NULL,@MaKH) select @MaDDH = max(idDonDatHang) from DonDatHang where idChiNhanh = @MaChiNhanh and idKhachHang = @MaKH insert into ChiTietDDH VALUES(@MaMon,@MaDDH,@SoLuongMua, @TuyChon) end			
B5: Cập nhật lại thông tin món ăn update MON set SoLuongHienTai = SoLuongHienTai - @SoLuongMua			

where idMon = @MaMon and idChiNhanh = @MaChiNhanh end			
		B4: Thêm món ăn vào đơn đặt hang insert into DonDatHang VALUES (N'Chờ nhận','2022-12-01', NULL,'Online', N'121/8 Phạm Văn Đồng, Quận Bình Thạnh', @Gia*@SoLuongMua, 3000, NULL, NULL, @MaChiNhanh, NULL, @MaKH) select @MaDDH = max(idDonDatHang) from DonDatHang where idChiNhanh = @MaChiNhanh and idKhachHang = @MaKH insert into ChiTietDDH VALUES(@MaMon,@MaDDH,@SoLuongMua, @TuyChon) end	
		B5: Cập nhật lại thông tin món ăn update MON set SoLuongHienTai = SoLuongHienTai - @SoLuongMua where idMon = @MaMon and idChiNhanh = @MaChiNhanh end	
COMMIT TRAN			
		COMMIT TRAN	

IV. Sinh viên thực hiện: Lâm Quang Duy

12. Tình huống 1: Chi nhánh xem các đơn đặt hàng để chọn lọc danh sách các đơn đặt hàng cần phải chuẩn bị. Cùng lúc đó khách hàng muốn huỷ đơn hàng ở trạng thái "chờ nhận" của mình bằng cách Xóa đơn đặt hàng. Chi nhánh xem lại đơn đặt hàng để gửi cho đầu bếp chuẩn bị các món ăn tuy nhiên không xem được nữa.

ERR0R1: *Unrepeatable Read*

T1 (User = Chi nhánh): thực hiện cập nhật thông tin đơn hàng A để nấu món ăn.

T2 (User = khách hàng): thực hiện xóa đơn hàng A của mình (kèm điều kiện tình trạng đơn hàng "Chờ nhận").

sp_ChiNhanhKiemTraDonHang	**Khóa**	**sp_KhachHangXoaDonHang**	**Khóa**
Input*: Mã chi nhánh, mã đơn đặt hàng.** ***Output: cập nhật đơn đặt hàng.		***Input***: ***Mã khách hàng, mã đơn hàng.*** ***Output***: Xóa đơn đặt hàng trên.	
SET TRANSACTION ISOLATION LEVEL ……		SET TRANSACTION ISOLATION LEVEL ……	
BEGIN TRAN			

B1: Chi nhánh kiểm tra các đơn đặt hàng
if(not exists(select * from DonDatHang where idChiNhanh = @MaCN))
begin			

raiserror('Chi nhánh không có đơn đặt hàng', 16, 1) rollback tran return end			
WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	

		B1: Kiểm tra tình trạng đơn hàng
		if((select TinhTrang from DonDatHang where idDonDatHang = @idDonDatHang) = N'Chờ nhận')
		begin
		delete ChiTietDDH where idDonDatHang = @idDonDatHang -- Do ràng buộc khóa chính khóa ngoại
		delete DonDatHang where idDonDatHang = @idDonDatHang	

		raiserror(N'Xóa thành công', 16,1)
		end
		else
		begin
		raiserror(N'Đơn hàng của bạn hiện không thể xóa', 16, 1)
		raiserror(N'Xóa không thành công', 16,1)
		rollback tran
		return
		end rollback tran
		return
		end	

		COMMIT TRAN	
B2: Chi nhánh lọc đơn hàng ở trạng thái "chờ nhận"			

select * from DonDatHang where TinhTrang = N'Chờ nhận' and idChiNhanh = @MaCN			
COMMIIT TRAN			

13. Tình huống 2: Tài xế nhận đơn đặt hàng và bấm xác nhận, chuyển đơn đặt hàng sang trạng thái "Đã nhận đơn hàng"., Chi nhánh kiểm tra đơn đặt hàng thấy hết món nên chuyển sang trạng thái huỷ đơn. Sau đó tài xế cập nhật tình trạng 'Đã nhận đơn hàng' ghi đè lên kết quả cập nhật của chi nhánh.

ERR0R2: *Lost Update* T1 (User = Khách hàng): thực hiện cập nhật thông tin đơn đặt hàng. T2 (User = Tài xế): thực hiện cập nhật thông tin đơn đặt hàng.			
sp_TaiXeCapNhatDonHang	**Khóa**	**sp_ChiNhanhCapNhatDonHang**	**Khóa**
Input*: Mã tài xế, mã đơn đặt hàng.** ***Output: Cập nhật đơn đặt hàng.		***Input***: ***Mã chi nhánh, mã đơn đặt hàng.*** ***Output***: Cập nhật đơn đặt hàng.	
SET TRANSACTION ISOLATION LEVEL ……		SET TRANSACTION ISOLATION LEVEL ……	
BEGIN TRAN			
B1: Kiểm tra tình trạng đơn hàng	T1 S(DonDatHang)		

if((select TinhTrang from DonDatHang where idDonDatHang = @idDonDatHang) = N'Chờ nhận')	T1 Xin được khóa S(Share) trên đơn đặt hàng và giữ đến cuối giao tác		
WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	

		B1: Kiểm tra đơn đặt hàng có thuộc chi nhánh
		
		if(not exists(select * from DonDatHang where idDonDatHang = @idDonDatHang and idChiNhanh = @MaCN))
		
		begin
		
		raiserror('Chi nhánh không có đơn đặt hàng này', 16, 1)
		
		raiserror(N'Cập nhật không thành công', 16,1)
		
		rollback tran	

		return end	

		B2: Chi nhánh huỷ đơn hàng
		
		if((select TinhTrang from DonDatHang where idDonDatHang = @idDonDatHang) = N'Chờ nhận')
		
		begin
		
		update DonDatHang set TinhTrang = N'Đã huỷ' where idDonDatHang = @idDonDatHang
		
		update DonDatHang set idChiNhanh = @MaCN where idDonDatHang = @idDonDatHang
		
		raiserror(N'Cập nhật thành công', 16,1)
		
		end
		
		else
		
		begin
			T2
			
			X(DonDatHang)
			
			T2 không thể xin được khóa X (Exclusive) trên DonDatHang do T1 đang giữ khóa S

		raiserror(N'Đơn hàng của bạn hiện không thể xóa', 16, 1) raiserror(N'Cập nhật không thành công', 16,1) rollback tran return end	
		COMMIT TRAN	
B2: Tài xế cập nhật trạng thái đơn hàng update DonDatHang set TinhTrang = N'Đã nhận đơn hàng' where idDonDatHang = @idDonDatHang update DonDatHang set idTaiXe = @MaTX where idDonDatHang = @idDonDatHang	T1 X(DonDatHang) T1 xin được khóa X (Exclusive) trên DonDatHang		
COMMIT TRAN			

Kết quả: Ban đầu tài xế thấy đơn hàng ở trạng thái "chờ nhận" thì xác nhận đơn hàng. Ngay lúc đó cửa hàng thấy hết món nên vào update huỷ đơn hàng. Bộ lập lịch cho phép cửa hàng huỷ đơn hàng, và chặn việc xác nhận đơn hàng của tài xế

14. Tình huống 3: Chi nhánh thống kê các đơn đặt hàng ở trạng thái "Chờ nhận. Ngay lúc đó khách hàng vào đặt thêm một vài đơn đặt hàng. Và sau đó, chi nhánh đưa danh sách đơn đặt hàng cho đầu bếp thực hiện thì thấy con số thống kê ở trên và số dòng mình in ra không khớp

ERR03: Phantom T1 (User = Chi nhánh): thực hiện thống kê số lượng đơn hàng và xem thông tin đơn đặt hàng. T2 (User = Khách hàng): thực hiện thêm đơn đặt hàng.			
sp_DoiTacThongKe	**Khóa**	**sp_KhachHangThemDongHang**	**Khóa**
Input*: Mã chi nhánh.** ***Output: Số lượng đơn hàng, danh sách đơn hàng.		***Input*: *Mã khách hàng, mã đơn đặt hàng, địa chỉ nhận, phí vận chuyển, hình thức thanh toán, mã chi nhánh, số lượng sản phẩm đặt, mã món.*** ***Output***: Thêm đơn đặt hàng.	
SET TRANSACTION ISOLATION LEVEL ……		SET TRANSACTION ISOLATION LEVEL ……	
BEGIN TRAN			

B1: Kiểm tra chi nhánh có tồn tại
 	
if(not exists(select * from ChiNhanh where idChiNhanh = @MaCN))
 	
begin
 	
raiserror(N'Mã chi nhánh không tồn tại', 16, 1)
 	
raiserror(N'Xóa không thành công', 16,1)
 	
rollback tran
 	
return
 	
end			

B2: Thống kê số lượng đơn đặt hàng ở trạng thái 'Chờ nhận' declare @SL_DonHang int set @SL_DonHang = (select count(*) from ChiNhanh CN, DonDatHang DDH where CN.idChiNhanh =			

DDH.idChiNhanh and DDH.TinhTrang = N'Chờ nhận')			
B3: In số lượng đơn đặt hàng print(N'Số lượng đơn hàng ở trạng thái chờ nhận của cửa hàng ' + cast(@MaCN as varchar(4)) + N' là: ' + cast(@SL_DonHang as varchar(4)))			
WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	

		B1: Kiểm tra thông tin của khách hàng
		
		if(not exists(select * from KhachHang where idKhachHang = @MaKH))
		begin
		raiserror(N'Mã khách hàng không tồn tại', 16, 1)
		raiserror(N'Xóa không thành công', 16,1)
		rollback tran	

		return end	

		B2: Kiểm tra đơn hàng có tồn tại
		if(exists(select * from DonDatHang where idDonDatHang = @idDonDatHang))
		begin
		raiserror(N'Đơn đặt hàng đã tồn tại', 16,1)
		rollback tran
		return
		end	

		B3: Kiểm tra chi nhánh có nằm trong hệ thống
		if(not exists(select * from ChiNhanh where idChiNhanh = @idChiNhanh))
		begin	

		raiserror(N'Chi nhánh không tồn tại trong hệ thống', 16,1) rollback tran return end	
		B4: Thêm đơn đặt hàng SET IDENTITY_INSERT DonDatHang ON insert into DonDatHang (idDonDatHang, TinhTrang, NgayLap, NgayThanhToan, HinhThucThanhToan, DiaChiNhan, PhiSanPham, PhiVanChuyen, Rating, Comment, idChiNhanh, idTaiXe, idKhachHang) values(@idDonDatHang, @TinhTrang, @NgayLap, @NgayThanhToan , @HinhThucThanhToan, @DiaChiNhan, @PhiSanPham, @PhiVanChuyen, @Rating, @Comment, @idChiNhanh, @idTaiXe, @MaKH)	

		insert into ChiTietDDH values(@MaMon, @idDonDatHang, @SLSanPham, @TuyChon)	
		COMMIT TRAN	
select DDH.idDonDatHang, DDH.TinhTrang, DDH.DiaChiNhan, DDH.HinhThucThanhToan, DDH.NgayLap, DDH.idTaiXe, DDH.idChiNhanh from ChiNhanh CN, DonDatHang DDH where CN.idChiNhanh = DDH.idChiNhanh and CN.idChiNhanh = @MaCN and DDH.TinhTrang = N'Chờ nhận'			
COMMIT TRAN			

GIẢI QUYẾT TÌNH HUỐNG TRANH CHẤP

I. Sinh viên thực hiện: Mai Nhật Nam

1. Tình huống 1: Tài xế nổ đơn thấy tình trạng đơn hàng đang là 'Chờ nhận', cùng lúc đó khách hàng muốn xóa đơn hàng của hàng của mình mà tình trạng là 'Chờ nhận' nên khách hàng click Xóa đơn đặt hàng. Điều này khiến cho tài xế không thể cập nhật để giao cho đơn đặt hàng này.

ERR01: ***Unrepeatable Read + Deadlock (Conversion Lock)***
T1 (User = tài xế): thực hiện cập nhật thông tin đơn hàng A để giao hàng.
T2 (User = khách hàng): thực hiện xóa đơn hàng A của mình (kèm điều kiện tình trạng đơn hàng "Chờ nhận").

SP_TAIXE_UPDATE_DONHANG	**Khóa**	**SP_KHACHHANG_DELETE_DONHANG**	**Khóa**
Input: Mã tài xế, mã đơn đặt hàng. ***Output***: cập nhật đơn đặt hàng.		***Input***: ***Mã khách hàng, mã đơn hàng.*** ***Output***: Xóa đơn đặt hàng trên.	
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ		SET TRANSACTION ISOLATION LEVEL REPEATABLE READ	
BEGIN TRAN			
B1: Kiểm tra tài xế có tồn tại IF NOT EXISTS (select * from TAI_XE where MaTX = TX001) begin print 'Tài xế không tồn tại' rollback tran return end			
B2: Kiểm tra thông tin đơn đặt hàng IF EXISTS (select * from DON_DAT_HANG where MaDDH = DH001) Begin Select * from DON_DAT_HANG where MaDDH = DH001	T1 S(DonDatHang) T1 Xin được khóa S(Share)		

	trên đơn đặt hàng và giữ đến cuối giao tác		
WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	
		B1: Kiểm tra khách hàng có tồn tại IF NOT EXISTS (select * from KHACH_HANG where KhachHang = KH002) Begin Print 'Khách hàng không tồn tại' Rollback tran Return End	T2 S(DonDatHang) T2 Xin được khóa S(Share) trên đơn đặt hàng và giữ đến cuối giao tác

		B2: Khách hàng DELETE đơn hàng
		
		IF (EXISTS (select * from DON_DAT_HANG where MaDDH = DH001 AND TinhTrang = 'Chờ nhận'))
		Begin
		DELETE DON_DAT_HANG
		Where MaDDH = DH001
		End
			T2 X(DonDatHang)
			
			T2 không thể xin được khóa X (Exclusive) trên DonDatHang do T1 đang giữ khóa S

		COMMIT TRAN	
B3: Kiểm tra tình trạng đơn hàng UPDATE DON_DAT_HANG set TinhTrang = 'Đã nhận' , MaTX = 2 where MaDDH = 1	T1 X(DonDatHang) **T1 không thể xin được khóa X (Exclusive)**		

End	**trên DonDatHang do T2 đang giữ khóa S**		
COMMIIT TRAN			

Kết quả: Nếu giải quyết 1 cách máy móc. Thấy bị Unrepeatable Read mà ta chỉ set Isolation ở mức Repeatable Read mà không quan tâm gì cả. Ta sẽ bị DEADLOCK và cụ thể ở trường hợp này cụ thể ta bị CONVERSION LOCK. Vì thế ta lại đặt một cái ROWLOCK, XLOCK ở lần đọc đầu tiên của giao tác Tài xế, tránh Share Lock và tránh trường hợp giao tác ta bị DEADLOCK.

ERR01: ***Unrepeatable Read + Deadlock***
T1 (User = tài xế): thực hiện cập nhật thông tin đơn hàng A để giao hàng.
T2 (User = khách hàng): thực hiện xóa đơn hàng A của mình (kèm điều kiện tình trạng đơn hàng "Chờ nhận").

SP_TAIXE_UPDATE_DONHANG	Khóa	SP_KHACHHANG_DELETE_DONHANG	Khóa
Input: Mã tài xế, mã đơn đặt hàng. ***Output***: cập nhật đơn đặt hàng.		***Input***: ***Mã khách hàng, mã đơn hàng.*** ***Output***: Xóa đơn đặt hàng trên.	
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ		SET TRANSACTION ISOLATION LEVEL REPEATABLE READ	
BEGIN TRAN			
B1: Kiểm tra tài xế có tồn tại IF NOT EXISTS (select * from TAI_XE where MaTX = TX001) begin print 'Tài xế không tồn tại' rollback tran return end			
B2: Kiểm tra thông tin đơn đặt hàng IF EXISTS (select * from DON_DAT_HANG WITH (ROWLOCK, XLOCK) where MaDDH = DH001)	T1 S(DonDatHang)		

Begin Select * from DON_DAT_HANG where MaDDH = DH001	T1 Xin được khóa S(Share) trên đơn đặt hàng và giữ đến cuối giao tác		
WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	
		B1: Kiểm tra khách hàng có tồn tại IF NOT EXISTS (select * from KHACH_HANG where KhachHang = KH002) Begin Print 'Khách hàng không tồn tại' Rollback tran Return End	T2 S(DonDatHang) T2 không xin được khóa S(Share) trên đơn đặt hàng và T1 có ROWLOCK, XLOCK

		B2: Khách hàng DELETE đơn hàng
		
		IF (EXISTS (select * from DON_DAT_HANG where MaDDH = DH001 AND TinhTrang = 'Chờ nhận'))
		Begin
		DELETE DON_DAT_HANG
		Where MaDDH = DH001
		End	

		COMMIT TRAN	
B3: Kiểm tra tình trạng đơn hàng UPDATE DON_DAT_HANG set TinhTrang = 'Đã nhận' , MaTX = 2 where MaDDH = 1	T1 X(DonDatHang) **T1 xin được khóa X**		

End	**(Exclusive) trên DonDatHang**		
COMMIIT TRAN			
Kết quả: Ta lại đặt ROWLOCK, XLOCK ở lần đọc đầu tiên đọc đầu tiên của giao tác Tài xế, tránh Share Lock cho giao tác Khách hàng. Mặc dù Share Lock có thể chia sẻ ở các giao tác nhưng do ta đã đặt ROWLOCK, XLOCK. Điều này giúp ta tránh trường hợp giao tác ta bị DEADLOCK.			

2. Tình huống 2: Tài xế nổ đơn hàng A nên chuẩn bị cập nhật để giao đơn hàng A, sau đó khách hàng muốn xóa đơn hàng A của mình nên đã vào cập nhật tình trạng đơn hàng A từ 'Chờ nhận' sang 'Đã hủy'. Sau đó tài xế cập nhật tình trạng 'Đã nhận đơn hàng' ghi đè lên kết quả cập nhật của khách hàng.

ERR01: ***Lost Update***
T1 (User = Khách hàng): thực hiện cập nhật thông tin đơn đặt hàng.
T2 (User = Tài xế): thực hiện cập nhật thông tin đơn đặt hàng.

SP_TAIXE_UPDATE_DONHANG	Khóa	sp_KhachHangCapNhatDonHang	Khóa
Input**: Mã tài xế, mã đơn đặt hàng.* ***Output: Cập nhật đơn đặt hàng.		***Input***: *Mã khách hàng, mã đơn đặt hàng.* ***Output***: Cập nhật đơn đặt hàng.	
SET TRANSACTION ISOLATION LEVEL READ COMMITTED		SET TRANSACTION ISOLATION LEVEL READ COMMITTED	
BEGIN TRAN			
B1: Kiểm tra thông tin tài xế IF NOT EXISTS (select * from TAI_XE where MaTX = TX001) Begin Print 'Tài xế không tồn tại' Rollback tran Return End			

B2: Kiểm tra thông tin đơn đặt hàng IF EXISTS (select * from DON_DAT_HANG where MaDDH = DH001) Begin Select * from DON_DAT_HANG where MaDDH = DH001 End	T1 S(DonDatHang) T1 Xin được khóa S(Share) trên đơn đặt hàng và Unlock		
WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	
		B1: Kiểm tra khách hàng có tồn tại IF NOT EXISTS (select * from KHACH_HANG where KhachHang = KH002) Begin Print 'Khách hàng không tồn tại' Rollback tran Return End	

		B2: Khách hàng UPDATE tình trạng
		
		IF (EXISTS (select * from DON_DAT_HANG where MaDDH = DH001 AND TinhTrang = 'Chờ nhận'))
		Begin
		UPDATE DON_DAT_HANG
		Set TinhTrang = 'Đã hủy'
		Where MaDDH = DH001
		End
			T2
			S(DonDatHang)
			
			T2 Xin được khóa S(Share) trên đơn đặt hàng và Unclock
			
			T2
			X(DonDatHang)
			
			T2 xin được khóa

			X(Exclusive) trên đơn đặt hàng và giữ khóa đến cuối giao tác
		COMMIT TRAN	

B3: Kiểm tra tình trạng đơn hàng
 	
IF (select TinhTrang from DON_DAT_HANG where MaDDH = DH001)
Begin
	Không thể vào IF do T2 đã cập nhật tình trạng		

UPDATE DON_DAT_HANG Set TinhTrang = 'Đã nhận đơn hàng', TaiXe = TX002 Where MaDDH = DH001 End	T1 X(DonDatHang) T1 xin được khóa X(Exclusive) trên đơn đặt hàng do T2 đã committ và giữ đến cuối giao tác		
COMMIT TRAN			
Kết quả: Ban đầu tài xế vào đọc thấy có đơn hàng. Sau đó khách hàng vào xem đơn hàng của mình và cập nhật tình trạng đơn hàng là 'Đã hủy'. Sau đó tài check tình trạng đơn hàng để nhận giao nhưng không được do khách hàng đã hủy => Cập nhật của khách hàng không bị mất.			

3. Tình huống 3: Cuối tháng đối tác vào thống kê xem tháng này trên hệ thống online mình bán được bao nhiêu bao nhiêu đơn hàng. Ngay sau đó khách hàng vào đặt thêm một vài đơn đặt hàng. Và sau đó, đối tác muốn in ra xem danh sách đơn đặt hàng thì thấy con số thống kê ở trên và số dòng mình in ra không khớp. Tại sao?

ERR01: Phantom
T1 (User = Đối tác): thực hiện thống kê số lượng đơn hàng và xem thông tin đơn đặt hàng.
T2 (User = Khách hàng): thực hiện thêm đơn đặt hàng.

sp_DoiTacThongKe	Khóa	sp_KhachHangInsertDDH	Khóa
Input**: Mã đối tác.* ***Output: Số lượng đơn hàng, danh sách đơn hàng.		***Input**: Mã khách hàng, mã đơn đặt hàng, địa chỉ nhận, phí vận chuyển, hình thức thanh toán, mã chi nhánh, số lượng sản phẩm đặt, mã món.* ***Output***: Thêm đơn đặt hàng.	
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE		SET TRANSACTION ISOLATION LEVEL READ COMMITED	
BEGIN TRAN			
B1: Kiểm tra thông tin đối tác IF NOT EXISTS (select * from DOI_TAC where MaDT = DT001) Begin Print 'Đối tác không tồn tại' Rollback tran Return End			
B2: Thống kê số lượng đơn đặt hàng Declare @SL_DonHang int Set @SL_DonHang = (select COUNT (*) From DOI_TAC DT, CHI_NHANH CN, DON_DAT_HANG DH Where DT. MaDT = DT001 and DT. MaDT=CN.MaDT and CN.MaCN = DH.MaCN)	T1 S(DonDatHang) T1 xin được khóa S (Share) trên đơn đặt hàng và giữ đến cuối giao tác.		

B3: In số lượng đơn đặt hàng Print 'Số lượng đơn hàng của đối tác' + DH001 + 'là' @SL_DonHang			
WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	
		B1: Kiểm tra thông tin của khách hàng IF NOT EXISTS (select * from KHACH_HANG where KhachHang= KH002) Begin Print 'Khách hàng không tông tại' Rollback tran Return End	
		B2: Kiểm tra mã đơn đặt hàng IF EXISTS (select * from DON_DAT_HANG where MaDDH = DH001) Begin PRINT 'Mã đơn hàng đã tồn tại' Set TinhTrang = 'Đã hủy' Where MaDDH = DH001 End	
		B3: Xử lý thông tin Decalre @DonGia float Declare @PhiSanPham Declare @ThanhTien Set @DonGia = (select Gia from MON where MaMon = @MaMon)	

		Set @PhiSanPham = @DonGia * SL_SanPham Set @ThanhTien @PhiSanPham + @PhiVanChuyen	
		B4: Thêm đơn đặt hàng INSERT INTO DON_DAT_HANG VALUES (@MaDDH, N'Chờ nhận', @DiaChiNhan, @PhiSanPham, @PhiVanChuyen, @ThanhTien, @HT_ThanhToan, GETDATE (), NULL, @MaKH, @MaChiNhanh) INSERT INTO CHI_TIET_DDH VALUES (@MaMon, @MaDDH, @SL_SanPham)	T2 X(DonDatHang) T2 không thể xin khóa X (Exclusive) trên đơn đặt hàng do T1 đang giữ khóa S. T2 không thể INSERT dòng dữ liệu mặc dù đã thỏa điều kiện.
		COMMIT TRAN	
B4: In danh sách đơn đặt hàng Select * From DOITAC DT, CHI_NHANH CN, DON_DAT_HANG DDH Where DT. MaDoiTac = @MaDT and DT. MaDoiTac = CN. DoiTac and CN. MaChiNhanh= DDH.ChiNhanh	T1 S(DonDatHang) T1 xin được khóa S (Share) trên đơn đặt hàng.		
COMMIT TRAN			

Kết quả: Khi tài xế thực hiện thống kê số lượng đơn hàng và xuất ra danh sách đơn hàng thì khách hàng không thể insert đơn hàng ngay lúc này. Tránh trường hợp thông tin trong báo cáo không đồng nhất. Khách hàng chỉ có thể thêm đơn hàng trước hoặc sau khi đối tác thực hiện thống kê.

4. Tình huống 4: Tại một thời điểm ta không 2 giao tác vào xem thuộc tính số lượng chi nhánh của tác ở hợp đồng tránh trường hợp cập nhật dữ liệu không đồng nhất và bị deadlock

ERR01: Cycle Lock
T1 (User = Nhân viên): thực hiện thêm chi nhánh và cập nhật ở hợp đồng.
T2 (User = Khách hàng): thực hiện xóa chi nhánh và cập nhật ở hợp đồng.

SP_NhanVienThemChiNhanh	Khóa	sp_KhachHangXoaChiNhanh	Khóa
Input**: Mã đối tác, tên chi nhánh, thời gian hoạt động, tình trạng, địa chỉ, số điện thoại.* ***Output: Thêm 1 chi nhánh cho đối tác và cập nhật số lượng chi nhánh bên hợp đồng của đối tác.		***Input***: *Mã đối tác, tên chi nhánh.* ***Output***: Xóa 1 chi nhánh cho đối tác	
SET TRANSACTION ISOLATION LEVEL READ COMMITED		SET TRANSACTION ISOLATION LEVEL READ COMMITED	
BEGIN TRAN			
B1: Kiểm tra thông tin đối tác IF NOT EXISTS (select * from DOI_TAC where MaDT = 1) Begin Print 'Đối tác không tồn tại' Rollback tran Return End			
B2: Xem số lượng chi nhánh đăng ký hiện tại của đối tác. Declare @SLChiNhanh int	T1 S(HopDong)		

Set @SL_DonHang = (select soluongChiNhanh From HOP_DONG Where DT. MaDT = 1)	T1 xin được khóa S trên Hợp đồng và sau đó nhả khóa.		
B3: In số lượng đơn đặt hàng Print 'Số lượng chi nhánh của đối tác trước khi thêm là' + @SLChiNhanh			
WAITFOR DELAY '00:00:3'			
		BEGIN TRAN	
		B1: Kiểm tra thông tin đối tác IF NOT EXISTS (select * from DOI_TAC where MaDT = 1) Begin Print 'Đối tác không tồn tại' Rollback tran Return End	
		B2: Xem số lượng chi nhánh đăng ký hiện tại của đối tác. Declare @SLChiNhanh int Set @SL_DonHang = (select soluongChiNhanh From HOP_DONG Where DT. MaDT = 1)	T2 S(HopDong) T2 xin được khóa S trên Hợp đồng và sau đó nhả khóa.

		B3: In số lượng đơn đặt hàng Print 'Số lượng chi nhánh của đối tác trước khi thêm là' + @SLChiNhanh	
		WAITFOR DELAY '00:00:3'	
B4: Thêm chi nhánh cho đối tác Insert into ChiNhanh values (TenChiNhanh, ThoiGianHD, TinhTrang, DiaChi, Sdt, idDoiTac)	T1 X(ChiNhanh) T1 xin được khóa X trên Chi Nhánh và giữ đến cuối giao tác.		
WAITFOR DELAY '00:00:3'			
		B4: Cập nhật số lượng chi nhánh đăng ký Update HOP_DONG set SoLuongCNDangKy = @SLChiNhanh – 1 where idDoiTac = 1	T2 X(HopDong) T2 xin được khóa X trên Hợp đồng và giữ đến cuối giao tác.
		WAITFOR DELAY '00:00:3'	
B5: Cập nhật số lượng chi nhánh đăng ký Update HOP_DONG set SoLuongCNDangKy = @SLChiNhanh + 1 where idDoiTac = 1	T1 X(HopDong) T1 không xin được khóa X trên Hợp đồng do T2 đang giữ khóa độc quyền.		
COMMIT TRAN			
		B5: Xóa chi nhánh	T2 X(ChiNhanh)

		Delete ChiNhanh where TenChiNhanh = 'Chi nhánh 4' and idDoiTac = 1	T2 không xin được khóa X trên Chi nhánh do T1 đang giữ khóa độc quyền.
		COMMIT TRAN	
Kết quả: T1 và T2 chờ nhau dẫn đến DEADLOCK và cụ thể là CYCLE LOCK. Giải quyết: Ta đặt (ROWLOCK, XLOCK) ngay lần đầu tiên đọc số lượng chi nhánh đăng ký trên hợp đồng của đối tác. Vừa tránh được tại một thời điểm 2 giao tác vào đọc và cũng giải quyết được DEADLOCK.			

II. Sinh viên thực hiện: Nguyễn Đặng Nam Khánh

5. Tình huống 1: Tài xế đầu tiên thấy tình trạng đơn hàng là chờ nhận nên chọn đơn hàng để giao, cùng lúc đó tài xế khác cũng thấy tình trạng đơn hàng là chờ nhận nên cũng chọn để giao. Hệ thống sẽ thông báo cả cho cả 2 là đều nhận đơn hàng thành công nhưng thực ra chỉ có tài xế 1 là nhận được đơn hàng.

ERROR 1: Lost Update T1(User=Tài xế): thực hiện chỉnh sửa đơn đặt hàng T2(User=Tài xế): thực hiện chỉnh sửa đơn đặt hàng			
USP_SUADDH_TAIXE1	**Khóa**	**USP_SUADDH_TAIXE2**	**Khóa**
Input*: Mã tài xế, mã đơn đặt hàng** ***Output: *Tài xế nhận được đơn đặt hàng*		***Input*: Mã tài xế, mã đơn đặt hàng** ***Output***: *Tài xế nhận được đơn đặt hàng*	
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ		SET TRANSACTION ISOLATION LEVEL REPEATABLE READ	
BEGIN TRAN			
B1: Kiểm tra tài xế có tồn tại			

IF (NOT EXISTS(select* from TaiXe where idTaiXe=@MaTaiXe))
BEGIN
print N'Tài xế không tồn tại'
rollback tran
return
END			

B2: Kiểm tra đơn đặt hàng
IF (EXISTS (select * from DonDatHang where idDonDatHang=@MaDDH and idTaiXe IS NULL and TinhTrang=N'Chờ nhận'))
BEGIN
	T1 được cấp khóa S trên DonDatHang và giữ đến hết giao tác		

WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	

		B1: Kiểm tra tài xế có tồn tại
		
		IF (NOT EXISTS(select* from TaiXe where idTaiXe=@MaTaiXe))
		BEGIN
		print N'Tài xế không tồn tại'
		rollback tran
		return
		END	

		B2: Kiểm tra đơn đặt hàng IF (EXISTS (select * from DonDatHang where idDonDatHang=@MaDDH and idTaiXe IS NULL and TinhTrang=N'Chờ nhận'))	T2 được cấp khóa S trên DonDatHang
B3: Cập nhật đơn đặt hàng UPDATE DonDatHang	T1 xin được cấp khóa X nhưng không được do T2 đang giữ khóa S		

SET TinhTrang=N'Đã nhận', idTaiXe=@MaTaiXe where idDonDatHang=@MaDDH print N'Nhận đơn hàng thành công' END end			
		B3: Cập nhật đơn đặt hàng UPDATE DonDatHang SET TinhTrang=N'Đã nhận', idTaiXe=@MaTaiXe where idDonDatHang=@MaDDH print N'Nhận đơn hàng thành công' END	T2 xin được cấp khóa X nhưng không được do T1 đang giữ khóa S
COMMIIT TRAN			
		COMMIT TRAN	
Kết quả: Giải quyết được lost update nhưng T1 thì thực hiện thành công nhưng T2 bị lỗi deadlock.			

6. Tình huống 2: Chi nhánh thứ nhất sửa món ăn, cùng lúc đó chi nhánh thứ 2 thực hiện xóa món ăn đó làm thao tác sửa món ăn của chi nhánh đầu tiên không thành công.

ERROR 2: Unrepeatable read **T1(User=Chi nhánh): thực hiện chỉnh sửa món** **T2(User=Chi nhánh): thực hiện xóa món**			
USP_SUAMON_CHINHANH	**Khóa**	USP_XOAMON_CHINHANH	**Khóa**

Input: ***Mã món, tên món mới, giá món mới, tình trạng*** ***Output****: Món đã thay đổi*		***Input****: **Mã món, mã chi nhánh*** ***Output****: Xóa món*
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ		SET TRANSACTION ISOLATION LEVEL READ COMMITTED
BEGIN TRAN		

B1: Kiểm tra chi nhánh có tồn tại hay không
IF (NOT EXISTS(select* from ChiNhanh where idChiNhanh=@MaCN))
BEGIN
print N'Chi nhánh không tồn tại'
rollback tran
return
END		

B2: Kiểm tra món có tồn tại hay không
IF (EXISTS (select * from Mon where idMon=@MaMon and idChiNhanh=@MaCN))
BEGIN
	T1 được cấp khóa S trên món và giữ đến hết giao tác	

WAITFOR DELAY '00:00:10'		
		BEGIN TRAN

		B1: Kiểm tra chi nhánh có tồn tại hay không
		IF (NOT EXISTS(select* from ChiNhanh where idChiNhanh=@MaCN))
		BEGIN
		print N'Chi nhánh không tồn tại'
		rollback tran
		return
		END

		B2: Xóa món
		IF (EXISTS (select * from Mon where idMon=@MaMon and idChiNhanh=@MaCN))
		BEGIN
		DELETE from Mon where idMon=@MaMon
		print N'Xóa món thành công'
		select * from Mon
		END
			T2 xin khóa S trên món và đọc xong thì trả lại
			
			T2 xin khóa X để xóa nhưng không được do T1 đang giữ khóa S trên món

B3: Cập nhật thông tin cho món UPDATE Mon SET TenMon=@TenMon,DonGia=@Gia,TinhTrang=@TinhTrang where idMon=@MaMon print N'Sửa món thành công' select * from Mon where idMon=@MaMon END	T1 xin được khóa X trên Món để cập nhật		
		COMMIT TRAN	
COMMIT TRAN			
Kết quả sau khi sửa lỗi: Chi nhánh thứ nhất sửa được món ăn xong chi nhánh thứ hai mới thực hiện được xóa món ăn			

7. Tình huống 3: Khách hàng muốn xem chi tiết một món ăn nhưng cùng lúc đó chi nhánh xóa món ăn đó làm khách hàng không thể xem được chi tiết món ăn

ERROR 3: Unrepeatable read
T1(User=Khách hàng): thực hiện xem món
T2(User=Chi nhánh): thực hiện xóa món

USP_XEMMON_KHACHHANG	Khóa	USP_XOAMON_CHINHANH	Khóa
***Input**: Mã món, mã chi nhánh* ***Output**: Chi tiết món ăn*		***Input**: Mã món, mã chi nhánh* ***Output**: Xóa món*	
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ		SET TRANSAVTION ISOLATION LEVEL READ COMMITTED	
BEGIN TRAN			

B1: Kiểm tra chi nhánh có tồn tại hay không
IF (NOT EXISTS(select* from ChiNhanh where idChiNhanh=@MaCN))
BEGIN
print N'Chi nhánh không tồn tại'
rollback tran
return
END			

B2: Kiểm tra món có tồn tại hay không
IF (EXISTS (select * from Mon where idMon=@MaMon and idChiNhanh=@MaCN))
BEGIN
	T1 được cấp khóa S trên món và giữ đến hết giao tác		

WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	

		B1: Kiểm tra chi nhánh có tồn tại hay không
		IF (NOT EXISTS(select* from ChiNhanh where idChiNhanh=@MaCN))
		BEGIN
		print N'Chi nhánh không tồn tại'	

		rollback tran return END	

		B2: Xóa món
		IF (EXISTS (select * from Mon where idMon=@MaMon and idChiNhanh=@MaCN))
		BEGIN
		DELETE from Mon where idMon=@MaMon
		print N'Xóa món thành công'
		select * from Mon
		END
			T2 xin khóa S trên món và đọc xong thì trả lại
			
			T2 xin khóa X để xóa nhưng không được do T1 đang giữ khóa S trên món

B3: Xuất thông tin món declare @ten nvarchar(50),@gia real,@tinhtrang nvarchar(50) select @ten=TenMon,@gia=DonGia,@tinhtrang=TinhTrang from Mon where idMon=@MaMon print N'Tên món: '+@ten print N'Giá: '+@gia print N'Tình trạng: '+@tinhtrang	T1 xin được khóa S trên món để xem chi tiết món ăn		
		COMMIT TRAN	
COMMIT TRAN			

Kết quả: Khách hàng xem được món ăn sau đó chi nhánh mới thực hiện xóa món ăn đó.

8. Tình huống 4: Tài xế cập nhật dữ liệu cho đơn hàng nhưng dữ liệu đơn hàng đưa vào không hợp lệ nên hệ thống rollback giao tác trên, cùng lúc đó thì khách hàng xem dữ liệu đơn hàng sẽ bị xem sai dữ liệu.

ERROR 4: Dirty Read **T1(User=Tài xế): thực hiện sửa đơn đặt hàng** **T2(User=Khách hàng): thực hiện xem món**			
USP_SUADDH_TAIXE	**Khóa**	USP_XEMMON_KHACHHANG	**Khóa**
Input*: Mã đơn đặt hàng, mã tài xế, tình trạng đơn đặt hàng** ***Output: Cập nhật dữ liệu cho đơn đặt hàng		***Input***: **Mã đơn đặt hàng, mã khách hàng** ***Output***: Chi tiết đơn đặt hàng	
SET TRANSACTION ISOLATION LEVEL READ COMMITTED		SET TRANSAVTION ISOLATION LEVEL READ COMMITTED	
BEGIN TRAN			

B1: Kiểm tra tài xế có tồn tại hay không
IF (NOT EXISTS(select* from TaiXe where idTaiXe=@MaTaiXe))
BEGIN
print N'Tài xế không tồn tại'
rollback tran
return
END			

B2: Cập nhật tình trạng đơn đặt hàng UPDATE DonDatHang SET TinhTrang=@TinhTrang, idTaiXe=@MaTaiXe where idDonDatHang=@MaDDH	T1 ghi mà không cần cấp khóa ghi		

print N'Nhận đơn hàng thành công' select * from DonDatHang where idDonDatHang=@MaDDH			
WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	

		IF (NOT EXISTS(select* from KhachHang where idKhachHang=@MaKH))
		BEGIN
		print N'Khách hàng không tồn tại'
		rollback tran
		return
		END	

		B2: Xuất thông tin đơn đặt hàng declare @ma nvarchar(50),@gia real,@tinhtrang nvarchar(20) select @ma=idDonDatHang,@gia=PhiSanPham+PhiVanChuyen,@tinhtrang=TinhTrang from DonDatHang where idDonDatHang=@MaDDH print N'Mã đơn đặt hàng: '+@ma print N'Giá: '+ CAST (@gia as varchar(20)) print N'Tình trạng: '+@tinhtrang	T2 đọc mà không cấp khóa đọc
IF EXISTS (select * from DonDatHang where TinhTrang NOT IN (N'Đã nhận',N'Đang giao',N'Chờ duyệt',N'Đang chờ')) BEGIN rollback tran	T1 rollback		

END			
		COMMIT TRAN	
COMMIT TRAN			

Kết quả: Khách hàng đợi đến khi tài xế ghi xong mới được đọc nên đọc được dữ liệu đúng.

III. Sinh viên thực hiện: Minh Triết

9. Tình huống 1: Chi nhánh 1 thay đổi thông tin món ăn, cùng lúc chi nhánh 2 cũng thay đổi thông tin của món ăn đó dẫn đến lost update.

ERR01: Lost Update
T1 (User = Chi Nhánh): thực hiện cập nhật thông tin món ăn
T2 (User = Chi Nhánh): thực hiện cập nhật thông tin món ăn

sp_SuaMonAn_ChiNhanh1	Khóa	sp_SuaMonAn_ChiNhanh2	Khóa
Input: Mã Món ăn, Mã chi nhánh ***Output: cập nhật món ăn***		***Input: Mã món ăn, Mã chi nhánh*** ***Output: Cập nhật thông tin món ăn***	
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ		SET TRANSACTION ISOLATION LEVEL REPEATABLE READ	
BEGIN TRAN			
B1: Kiểm tra món ăn có tồn tại if not exists (select* from MON where idMon = @MaMon and idChiNhanh = @MaChiNhanh) begin print N'Món ăn này không tồn tại!' rollback tran return end	T1 được cấp khóa S trên MON đến hết giao tác		
B2: Nếu món ăn có tồn tại			

if exists (select * from MON where idMon = @MaMon and idChiNhanh = @MaChiNhanh) begin			
WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	
		B1: Kiểm tra món ăn có tồn tại if not exists (select* from MON where idMon = @MaMon and idChiNhanh = @MaChiNhanh) begin print N'Món ăn này không tồn tại!' rollback tran return end	T2 được cấp khóa S trên MON đến hết giao tác
		B2: Nếu món ăn có tòn tại if exists (select * from MON where idMon = @MaMon and idChiNhanh = @MaChiNhanh) begin	
B3: Cập nhật lại món ăn update MON set MieuTa = @MieuTa where idMon = @MaMon and TenMon = @TenMon and idChiNhanh = @MaChiNhanh update MON set DonGia = @Gia where idMon = @MaMon and TenMon = @TenMon and idChiNhanh = @MaChiNhanh	T1 xin cấp khóa X nhưng không được do T2 đang giữ khóa S		

print N'Sửa món ăn thành công' select * from MON where idMon = @MaMon and idChiNhanh = @MaChiNhanh end			
		B3: Cập nhật lại món ăn update MON set MieuTa = @MieuTa where idMon = @MaMon and TenMon = @TenMon and idChiNhanh = @MaChiNhanh update MON set DonGia = @Gia where idMon = @MaMon and TenMon = @TenMon and idChiNhanh = @MaChiNhanh print N'Sửa món ăn thành công' select * from MON where idMon = @MaMon and idChiNhanh = @MaChiNhanh end	T2 xin cấp khóa X nhưng không được do T1 đang giữ khóa S
COMMIIT TRAN			
		COMMIIT TRAN	
Kết quả: Giải quyết được lost update nhưng T2 thực hiện thành công còn T1 thì bị deadlock			

10. Tình huống 2: Nhân viên 1 thay đổi ngày kết thúc của hợp đồng cùng lúc đó nhân viên 2 cũng vào thay đổi ngày kết thúc của hợp đồng đó dẫn đến Lost update

ERR02: Lost Update
T1 (User = Nhân viên): thực hiện thay đổi ngày kết thúc của hợp đồng
T2 (User = Nhân viên) : thực hiện thay đổi ngày kết thúc của hợp đồng

sp_ChinhSuaHD_NhanVien1	Khóa	sp_ChinhSuaHD_NhanVien2	Khóa
Input**: Mã hợp đồng, mã nhân viên, mã đối tác* ***Output: Cập nhật hợp đồng		***Input***: *Mã hợp đồng, mã nhân viên, mã đối tác.* ***Output***: Cập nhật hợp đồng	
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ		SET TRANSACTION ISOLATION LEVEL REPEATABLE READ	
BEGIN TRAN			
B1: Kiểm tra thông tin hợp đồng if not exists (select * from HOPDONG where idHopDong = @MaHD and idNhanVien = @MaNv and idDoiTac = @MaDoiTac) begin print N'Hợp đồng này không tồn tại' rollback tran return end	T1 được cấp khóa S trên HOPDONG đến hết giao tác		
B2: Nếu hợp đồng có tồn tại if exists (select * from HOPDONG where idHopDong = @MaHD and idNhanVien = @MaNv and idDoiTac = @MaDoiTac) begin			
WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	
		B1: Kiểm tra hợp đồng	T2 được cấp khóa S trên HOPDONG

		if not exists (select * from HOPDONG where idHopDong = @MaHD and idNhanVien = @MaNv and idDoiTac = @MaDoiTac) begin print N'Hợp đồng này không tồn tại' rollback tran return end	đến hết giao tác
		B2: Nếu hợp đồng tồn tại if exists (select * from HOPDONG where idHopDong = @MaHD and idNhanVien = @MaNv and idDoiTac = @MaDoiTac) begin	
B3: Tiến hành cập nhật lại hợp đồng update HOPDONG set ThoiHan = @ThoiHan where idHopDong = @MaHD and idNhanVien = @MaNv and idDoiTac = @MaDoiTac end	T1 xin cấp khóa X nhưng không được do T2 đang giữ khóa S		
		B3: Tiến hành cập nhật lại hợp đồng update HOPDONG set ThoiHan = @ThoiHan where idHopDong = @MaHD and idNhanVien = @MaNv and idDoiTac = @MaDoiTac end	T2 xin cấp khóa X nhưng không được do T1 đang giữ khóa S
COMMIT TRAN			
		COMMIT TRAN	

Kết quả: Giải quyết được lost update nhưng T2 thực hiện thành công còn T1 thì bị deadlock

11. Tình huống 3: Khánh hàng 1 vào xem và đặt món ăn, cùng lúc đó khách hàng 2 cũng vào xem và đặt món ăn đó làm cho cho việc xem món ăn của khách hàng 1 bị sai gây ra LOST UPDATE

ERR03: Lost Update
T1 (User = Khách Hàng 1): thực hiện đặt món ăn
T2 (User = Khách Hàng 2): thực hiện đặt món ăn

sp_DatMonAn_KhachHang1	Khóa	sp_DatMonAn_KhachHang2	Khóa
Input**: Mã món ăn, Mã Chi Nhánh, Mã Khách Hàng, Số lượng đặt của món đó* ***Output: Thêm đơn đặt hàng		***Input***: *Mã món ăn, Mã Chi Nhánh, Mã Khách Hàng, Số lượng đặt của món đó* ***Output***: Thêm đơn đặt hàng.	
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ		SET TRANSACTION ISOLATION LEVEL REPEATABLE READ	
BEGIN TRAN			
B1: Kiểm tra thông tin món ăn if not exists (select* from MON where idMon = @MaMon and idChiNhanh = @MaChiNhanh) begin print N'Món ăn này không tồn tại!' rollback tran return end	T1 được cấp khóa S trên MON đến hết giao tác		
B2: Nếu món ăn tồn tại if exists (select* from MON where idMon = @MaMon and idChiNhanh = @MaChiNhanh)			

begin			
B3: Kiểm tra số lượng món ăn begin declare @Gia real, @SLHT int, @MaDDH int select @Gia = DonGia, @SLHT = SoLuongHienTai from MON where idMon = @MaMon			
WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	
		B1: Kiểm tra thông tin món ăn if not exists (select* from MON where idMon = @MaMon and idChiNhanh = @MaChiNhanh) begin print N'Món ăn này không tồn tại!' rollback tran return end	T2 được cấp khóa S trên MON đến hết giao tác
		B2: Nếu món ăn tồn tại if exists (select* from MON where idMon = @MaMon and idChiNhanh = @MaChiNhanh) begin	
		B3: Kiểm tra số lượng món ăn begin declare @Gia real, @SLHT int, @MaDDH int select @Gia = DonGia, @SLHT = SoLuongHienTai from MON where idMon = @MaMon	

B4: Thêm món ăn vào đơn đặt hang insert into DonDatHang VALUES(N'Chờ nhận',GETDATE(),NULL,NULL,@DiaChi,@Gia*@SoLuongMua,3000,NULL,NULL,@MaChiNhanh,NULL,@MaKH) select @MaDDH = max(idDonDatHang) from DonDatHang set @SLHT=@SLHT-@SoLuongMua insert into ChiTietDDH VALUES(@MaMon,@MaDDH,@SoLuongMua,@TuyChon)	T1 xin cấp khóa X nhưng không được do T2 đang giữ khóa S		
B5: Cập nhật lại thông tin món ăn update MON set SoLuongHienTai = @SLHT where idMon = @MaMon and idChiNhanh = @MaChiNhanh end			
		B4: Thêm món ăn vào đơn đặt hang insert into DonDatHang VALUES(N'Chờ nhận',GETDATE(),NULL,NULL,@DiaChi,@Gia*@SoLuongMua,3000,NULL,NULL,@MaChiNhanh,NULL,@MaKH) select @MaDDH = max(idDonDatHang) from DonDatHang set @SLHT=@SLHT-@SoLuongMua	T2 xin cấp khóa X nhưng không được do T1 đang giữ khóa S

		insert into ChiTietDDH VALUES(@MaMon,@MaDDH,@SoLuongMua, @TuyChon)	
		B5: Cập nhật lại thông tin món ăn update MON set SoLuongHienTai = @SLHT where idMon = @MaMon and idChiNhanh = @MaChiNhanh end	
COMMIT TRAN			
		COMMIT TRAN	
Kết quả: Giải quyết được lost update nhưng T2 thực hiện thành công còn T1 thì bị deadlock			

IV. Sinh viên thực hiện: Lâm Quang Duy

12. Tình huống 1: Chi nhánh xem các đơn đặt hàng để chọn lọc danh sách các đơn đặt hàng cần phải chuẩn bị. Cùng lúc đó khách hàng muốn huỷ đơn hàng ở trạng thái "chờ nhận" của mình bằng cách Xóa đơn đặt hàng. Chi nhánh xem lại đơn đặt hàng để gửi cho đầu bếp chuẩn bị các món ăn tuy nhiên không xem được nữa.

ERR01: ***Unrepeatable Read***
T1 (User = Chi nhánh): thực hiện cập nhật thông tin đơn hàng A để nấu món ăn.
T2 (User = khách hàng): thực hiện xóa đơn hàng A của mình (kèm điều kiện tình trạng đơn hàng "Chờ nhận").

sp_ChiNhanhKiemTraDonHang	Khóa	sp_KhachHangXoaDonHang	Khóa
Input: Mã chi nhánh, mã đơn đặt hàng. ***Output***: cập nhật đơn đặt hàng.		***Input***: ***Mã khách hàng, mã đơn hàng.*** ***Output***: Xóa đơn đặt hàng trên.	
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ		SET TRANSACTION ISOLATION LEVEL REPEATABLE READ	
BEGIN TRAN			

B1: Chi nhánh kiểm tra các đơn đặt hàng
 	
if(not exists(select * from DonDatHang where idChiNhanh = @MaCN))
begin
raiserror('Chi nhánh không có đơn đặt hàng', 16, 1)
rollback tran
return
end
	T1
	S(DonDatHang)
	
	T1 Xin được khóa S(Share) trên đơn đặt hàng và giữ đến cuối giao tác		

WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	

		B1: Kiểm tra tình trạng đơn hàng
		
		if((select TinhTrang from DonDatHang where idDonDatHang = @idDonDatHang) = N'Chờ nhận')
		begin
		delete ChiTietDDH where idDonDatHang = @idDonDatHang -- Do ràng buộc khóa chính khóa ngoại
		delete DonDatHang where idDonDatHang = @idDonDatHang
		raiserror(N'Xóa thành công', 16,1)
		end
		else
		begin
		raiserror(N'Đơn hàng của bạn hiện không thể xóa', 16, 1)
			T2
			X(DonDatHang)
			
			T2 không thể xin được khóa X (Exclusive) trên DonDatHang do T1 đang giữ khóa S

		raiserror(N'Xóa không thành công', 16,1) rollback tran return end rollback tran return end	
		COMMIT TRAN	
B2: Chi nhánh lọc đơn hàng ở trạng thái "chờ nhận" select * from DonDatHang where TinhTrang = N'Chờ nhận' and idChiNhanh = @MaCN	T1 S(DonDatHang) T1 xin được khóa S (Share) trên đơn đặt hàng.		
COMMIIT TRAN			
Kết quả: Ban đầu chi nhánh xem đơn hàng, sau đó khách hàng gửi yêu cầu xoá đơn hàng. Yêu cầu xoá đơn hàng của khách hàng được bộ lập lịch cho thực hiện sau khi chi nhánh hoàn thành việc xem đơn hàng, nên dữ liệu chi nhánh xem được không thay đổi (có thể đọc lại dữ liệu).			

13. Tình huống 2: Tài xế nhận đơn đặt hàng và bấm xác nhận, chuyển đơn đặt hàng sang trạng thái "Đã nhận đơn hàng"., Chi nhánh kiểm tra đơn đặt hàng thấy hết món nên chuyển sang trạng thái huỷ đơn. Sau đó tài xế cập nhật tình trạng 'Đã nhận đơn hàng' ghi đè lên kết quả cập nhật của chi nhánh.

ERR02: *Lost Update* T1 (User = Khách hàng): thực hiện cập nhật thông tin đơn đặt hàng. T2 (User = Tài xế): thực hiện cập nhật thông tin đơn đặt hàng.			
sp_TaiXeCapNhatDonHang	**Khóa**	**sp_ChiNhanhCapNhatDonHang**	**Khóa**

Input**: Mã tài xế, mã đơn đặt hàng.* ***Output: Cập nhật đơn đặt hàng.		***Input***: *Mã chi nhánh, mã đơn đặt hàng.* ***Output***: Cập nhật đơn đặt hàng.
SET TRANSACTION ISOLATION LEVEL SNAPSHOT		SET TRANSACTION ISOLATION LEVEL SNAPSHOT
BEGIN TRAN		
B1: Kiểm tra tình trạng đơn hàng if((select TinhTrang from DonDatHang where idDonDatHang = @idDonDatHang) = N'Chờ nhận')	T1 S(DonDatHang) T1 Xin được khóa S(Share) trên đơn đặt hàng và giữ đến cuối giao tác	
WAITFOR DELAY '00:00:10'		
		BEGIN TRAN

		B1: Kiểm tra đơn đặt hàng có thuộc chi nhánh
		if(not exists(select * from DonDatHang where idDonDatHang = @idDonDatHang and idChiNhanh = @MaCN))
		begin
		raiserror('Chi nhánh không có đơn đặt hàng này', 16, 1)
		raiserror(N'Cập nhật không thành công', 16,1)
		rollback tran
		return
		end

		B2: Chi nhánh huỷ đơn hàng
		
		if((select TinhTrang from DonDatHang where idDonDatHang = @idDonDatHang) = N'Chờ nhận')
		begin
		update DonDatHang set TinhTrang = N'Đã huỷ' where idDonDatHang = @idDonDatHang
		update DonDatHang set idChiNhanh = @MaCN where idDonDatHang = @idDonDatHang
		raiserror(N'Cập nhật thành công', 16,1)
		end
		else
		begin
		raiserror(N'Đơn hàng của bạn hiện không thể xóa', 16, 1)
		raiserror(N'Cập nhật không thành công', 16,1)
		rollback tran
		return
		end
			T2
			X(DonDatHang)
			
			T2 không thể xin được khóa X (Exclusive) trên DonDatHang do T1 đang giữ khóa S

		COMMIT TRAN	
B2: Tài xế cập nhật trạng thái đơn hàng update DonDatHang set TinhTrang = N'Đã nhận đơn hàng' where idDonDatHang = @idDonDatHang update DonDatHang set idTaiXe = @MaTX where idDonDatHang = @idDonDatHang	T1 X(DonDatHang) T1 xin được khóa X		

	(Exclusive) trên DonDatHang		
COMMIT TRAN			
Kết quả: Ban đầu tài xế thấy đơn hàng ở trạng thái "chờ nhận" thì xác nhận đơn hàng. Ngay lúc đó cửa hàng thấy hết món nên vào update huỷ đơn hàng. Bộ lập lịch cho phép cửa hàng huỷ đơn hàng, và chặn việc xác nhận đơn hàng của tài xế			

14. Tình huống 3: Chi nhánh thống kê các đơn đặt hàng ở trạng thái "Chờ nhận. Ngay lúc đó khách hàng vào đặt thêm một vài đơn đặt hàng. Và sau đó, chi nhánh đưa danh sách đơn đặt hàng cho đầu bếp thực hiện thì thấy con số thống kê ở trên và số dòng mình in ra không khớp

ERR03: Phantom T1 (User = Chi nhánh): thực hiện thống kê số lượng đơn hàng và xem thông tin đơn đặt hàng. T2 (User = Khách hàng): thực hiện thêm đơn đặt hàng.			
sp_DoiTacThongKe	**Khóa**	**sp_KhachHangThemDongHang**	**Khóa**
Input**: Mã chi nhánh.* ***Output: Số lượng đơn hàng, danh sách đơn hàng.		***Input**: Mã khách hàng, mã đơn đặt hàng, địa chỉ nhận, phí vận chuyển, hình thức thanh toán, mã chi nhánh, số lượng sản phẩm đặt, mã món.* ***Output***: Thêm đơn đặt hàng.	
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE		SET TRANSACTION ISOLATION LEVEL SERIALIZABLE	
BEGIN TRAN			

B1: Kiểm tra chi nhánh có tồn tại
 	
if(not exists(select * from ChiNhanh where idChiNhanh = @MaCN))
begin			

raiserror(N'Mã chi nhánh không tồn tại', 16, 1) raiserror(N'Xóa không thành công', 16,1) rollback tran return end			
B2: Thống kê số lượng đơn đặt hàng ở trạng thái 'Chờ nhận' declare @SL_DonHang int set @SL_DonHang = (select count(*) from ChiNhanh CN, DonDatHang DDH where CN.idChiNhanh = DDH.idChiNhanh and DDH.TinhTrang = N'Chờ nhận')	T1 S(DonDatHang) T1 xin được khóa S (Share) trên đơn đặt hàng và giữ đến cuối giao tác.		
B3: In số lượng đơn đặt hàng print(N'Số lượng đơn hàng ở trạng thái chờ nhận của cửa hàng ' + cast(@MaCN as varchar(4)) + N' là: ' + cast(@SL_DonHang as varchar(4))))			
WAITFOR DELAY '00:00:10'			
		BEGIN TRAN	

		B1: Kiểm tra thông tin của khách hàng
		
		if(not exists(select * from KhachHang where idKhachHang = @MaKH))
		begin
		raiserror(N'Mã khách hàng không tồn tại', 16, 1)	

		raiserror(N'Xóa không thành công', 16,1) rollback tran return end	

		B2: Kiểm tra đơn hàng có tồn tại
		
		if(exists(select * from DonDatHang where idDonDatHang = @idDonDatHang))
		begin
		raiserror(N'Đơn đặt hàng đã tồn tại', 16,1)
		rollback tran
		return
		end	

		B3: Kiểm tra chi nhánh có nằm trong hệ thống
		if(not exists(select * from ChiNhanh where idChiNhanh = @idChiNhanh))
		begin
		raiserror(N'Chi nhánh không tồn tại trong hệ thống', 16,1)
		rollback tran
		return
		end	

		B4: Thêm đơn đặt hàng SET IDENTITY_INSERT DonDatHang ON insert into DonDatHang (idDonDatHang, TinhTrang, NgayLap, NgayThanhToan, HinhThucThanhToan, DiaChiNhan,	T2 X(DonDatHang) T2 không thể xin khóa X

		PhiSanPham, PhiVanChuyen, Rating, Comment, idChiNhanh, idTaiXe, idKhachHang) values(@idDonDatHang, @TinhTrang, @NgayLap, @NgayThanhToan , @HinhThucThanhToan, @DiaChiNhan, @PhiSanPham, @PhiVanChuyen, @Rating, @Comment, @idChiNhanh, @idTaiXe, @MaKH) insert into ChiTietDDH values(@MaMon, @idDonDatHang, @SLSanPham, @TuyChon)	(Exclusive) trên đơn đặt hàng do T1 đang giữ khóa S. T2 không thể INSERT dòng dữ liệu mặc dù đã thỏa điều kiện.
		COMMIT TRAN	
select DDH.idDonDatHang, DDH.TinhTrang, DDH.DiaChiNhan, DDH.HinhThucThanhToan, DDH.NgayLap, DDH.idTaiXe, DDH.idChiNhanh from ChiNhanh CN, DonDatHang DDH where CN.idChiNhanh = DDH.idChiNhanh and CN.idChiNhanh = @MaCN and DDH.TinhTrang = N'Chờ nhận'	T1 S(DonDatHang) T1 xin được khóa S (Share) trên đơn đặt hàng.		
COMMIT TRAN			
Kết quả: Ban đầu chi nhánh vào thống kê số lượng đơn hàng ở trạng thái "chờ nhận" của mình. Ngay lúc đó, khách hàng vào đặt thêm đơn đặt hàng. Bộ lập lịch sắp xếp việc khách hàng thêm đơn hàng sau khi hệ thống thực hiện thống kê xong. Nên số lượng đơn hàng thống kê và số lượng đơn hàng in ra của hệ thống là giống nhau. Sau đó mới cho phép khách hàng thêm đơn đặt hàng vào.			