Purple--- I know why it’s not in but I could be wrong.

Pink--- I added it to connection page.

Green--- is the same

/\*

CNSA 216 Final Project

Stored Procedures

\*/

USE webprog\_mmn;

--\*\*\* Add records \*\*\*

--add client

**CREATE PROC addNewClient (**

**@fName VARCHAR(25),**

**@mInitial CHAR(1) = '',**

**@lName VARCHAR(25),**

**@street1 VARCHAR(100) = '',**

**@street2 VARCHAR(100) = '',**

**@city VARCHAR(25) = '',**

**@addr\_state VARCHAR(25) = '',**

**@zip VARCHAR(10) = '',**

**@phone VARCHAR(15) = '',**

**@email VARCHAR(100) = '',**

**@gender CHAR(1),**

**@dateOfBirth DATE**

**)**

**AS**

**IF NOT EXISTS (**

**SELECT 1 FROM client**

**WHERE UPPER(fName) = UPPER(@fName)**

**AND UPPER(lName) = UPPER(@lName)**

**AND dateOfBirth = @dateOfBirth**

**)**

**BEGIN**

**SET NOCOUNT ON;**

**BEGIN TRANSACTION**

**INSERT INTO client(fName, mInitial, lName, street1,**

**street2, city, addr\_state, zip, phone, email,**

**gender, dateOfBirth)**

**VALUES(@fName, @mInitial, @lName, @street1,**

**@street2, @city, @addr\_state, @zip, @phone, @email,**

**@gender, @dateOfBirth)**

**DECLARE @RESULT INT**

**IF @@ERROR <> 0**

**BEGIN**

**ROLLBACK TRANSACTION**

**SET @RESULT = -1**

**RETURN @RESULT**

**END**

**ELSE**

**BEGIN**

**COMMIT TRANSACTION**

**--get the just inserted client id**

**SET @RESULT = @@IDENTITY**

**--create record for clientInsurance**

**INSERT INTO clientInsurance(clientID, insuranceID, clientInsuranceNum)**

**VALUES(@RESULT, 1, @RESULT + 100)**

**RETURN @RESULT**

**END**

**END**

**ELSE**

**BEGIN**

**RETURN 0**

**END**

**CREATE PROC newClientRegistration(**

**@clientID INT,**

**@username VARCHAR(25),**

**@clientPassword NVARCHAR(MAX),**

**@comPassword NVARCHAR(MAX),**

**@salt NVARCHAR(512)**

**)**

**AS**

**BEGIN**

**SET NOCOUNT ON;**

**--insert username into client**

**BEGIN TRANSACTION**

**UPDATE client**

**SET username = @username**

**WHERE clientID = @clientID**

**IF @@ERROR <> 0**

**BEGIN**

**ROLLBACK TRANSACTION**

**RETURN**

**END**

**ELSE**

**BEGIN**

**COMMIT TRANSACTION**

**INSERT INTO users**

**VALUES(@clientID, @clientPassword, @comPassword, @salt, 1)**

**END**

**END**

**GO**

--add insurance company

CREATE PROC addNewInsurance (

@companyName VARCHAR(50),

@coveragePercent DECIMAL(5,2),

@phone VARCHAR(15) = '',

@street1 VARCHAR(100) = '',

@street2 VARCHAR(100) = '',

@city VARCHAR(25) = '',

@addr\_state VARCHAR(25) = '',

@zip VARCHAR(10) = ''

)

AS

IF NOT EXISTS (

SELECT 1 FROM insurance

WHERE UPPER(companyName) = UPPER(@companyName)

)

BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION

INSERT INTO insurance(companyName, coveragePercent,

phone, street1, street2, city, addr\_state, zip)

VALUES(@companyName, @coveragePercent, @phone,

@street1, @street2, @city, @addr\_state, @zip)

IF @@ERROR <> 0

BEGIN

ROLLBACK TRANSACTION

RAISERROR('Unable to insert record.',16,1)

RETURN -1

END

ELSE

BEGIN

COMMIT TRANSACTION

PRINT('Record added successfully!')

END

END

ELSE

BEGIN

PRINT('Record already exists.')

RETURN

END

GO

--add new client insurance record

CREATE PROC newClientInsuranceRecord (

@clientID INT,

@insuranceID INT,

@clientInsuranceNum INT

)

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION

INSERT INTO clientInsurance(clientID, insuranceID, clientInsuranceNum)

VALUES(@clientID, @insuranceID, @clientInsuranceNum)

IF @@ERROR <> 0

BEGIN

ROLLBACK TRANSACTION

RAISERROR('Unable to insert record.',16,1)

RETURN -1

END

ELSE

BEGIN

COMMIT TRANSACTION

PRINT('Record added successfully!')

END

END

GO

--new physician

**CREATE PROC addNewPhysician (**

**@fName VARCHAR(25),**

**@mInitial CHAR(1) = '',**

**@lName VARCHAR(25),**

**@phone VARCHAR(15) = '',**

**@email VARCHAR(100) = ''**

**)**

**AS**

**IF NOT EXISTS (**

**SELECT 1 FROM physician**

**WHERE UPPER(fName) = UPPER(@fName)**

**AND UPPER(lName) = UPPER(@lName)**

**)**

**BEGIN**

**SET NOCOUNT ON;**

**BEGIN TRANSACTION**

**INSERT INTO physician(fName, mInitial, lName,**

**phone, email)**

**VALUES(@fName, @mInitial, @lName, @phone, @email)**

**DECLARE @RESULT INT**

**IF @@ERROR <> 0**

**BEGIN**

**ROLLBACK TRANSACTION**

**SET @RESULT = -1**

**RETURN @RESULT**

**END**

**ELSE**

**BEGIN**

**COMMIT TRANSACTION**

**SET @RESULT = @@IDENTITY**

**RETURN @RESULT**

**END**

**END**

**ELSE**

**BEGIN**

**SET @RESULT = 0**

**RETURN @RESULT**

**END**

**GO**

--new medicine

CREATE PROC addNewMedicine (

@medicineName VARCHAR(25),

@intakeMethod VARCHAR(100),

@cost DECIMAL(7,2)

)

AS

IF NOT EXISTS (

SELECT 1 FROM medicine

WHERE UPPER(medicineName) = UPPER(@medicineName)

AND UPPER(intakeMethod) = UPPER(@intakeMethod)

)

BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION

INSERT INTO medicine(medicineName, intakeMethod, cost)

VALUES(@medicineName, @intakeMethod, @cost)

IF @@ERROR <> 0

BEGIN

ROLLBACK TRANSACTION

RAISERROR('Unable to insert record.',16,1)

RETURN -1

END

ELSE

BEGIN

COMMIT TRANSACTION

PRINT('Record added successfully!')

END

END

ELSE

BEGIN

PRINT('Record already exists.')

RETURN

END

GO

**--new prescription**

**CREATE PROC addNewPrescription (**

**@clientID INT,**

**@physicianID INT,**

**@medicineID INT,**

**@expiryDate DATE = '',**

**@refillCounter TINYINT = ''**

**)**

**AS**

**BEGIN**

**SET NOCOUNT ON;**

**BEGIN TRANSACTION**

**INSERT INTO prescription(clientID, physicianID, medicineID,**

**expiryDate, refillCounter)**

**VALUES(@clientID, @physicianID, @medicineID, @expiryDate, @refillCounter)**

**DECLARE @RESULT INT**

**IF @@ERROR <> 0**

**BEGIN**

**ROLLBACK TRANSACTION**

**SET @RESULT = -1**

**RETURN @RESULT**

**END**

**ELSE**

**BEGIN**

**COMMIT TRANSACTION**

**SET @RESULT = @@IDENTITY**

**RETURN @RESULT**

**END**

**END**

**GO**

**--new refill**

**CREATE PROC addNewRefill (**

**@prescriptionID INT,**

**@dosage VARCHAR(50),**

**@frequency VARCHAR(50),**

**@supplyDays TINYINT,**

**@quantitySupplied TINYINT**

**)**

**AS**

**BEGIN**

**SET NOCOUNT ON;**

**BEGIN TRANSACTION**

**INSERT INTO refill(prescriptionID, dosage, frequency, supplyDays,**

**quantitySupplied)**

**VALUES(@prescriptionID, @dosage, @frequency, @supplyDays,**

**@quantitySupplied)**

**DECLARE @RESULT INT**

**IF @@ERROR <> 0**

**BEGIN**

**ROLLBACK TRANSACTION**

**SET @RESULT = -1**

**RETURN @RESULT**

**END**

**ELSE**

**BEGIN**

**COMMIT TRANSACTION**

**SET @RESULT = @@IDENTITY**

**RETURN @RESULT**

**END**

**END**

**GO**

Since you can only pay in full?--new payment

CREATE PROC newPayment (

@refillID INT,

@paymentType VARCHAR(6)

)

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRANSACTION

INSERT INTO payment(refillID, paymentType)

VALUES(@refillID, UPPER(@paymentType))

IF @@ERROR <> 0

BEGIN

ROLLBACK TRANSACTION

RAISERROR('Unable to insert record.',16,1)

RETURN -1

END

ELSE

BEGIN

COMMIT TRANSACTION

PRINT('Record added successfully!')

END

END

GO

**CREATE PROC newEmployeeRegistration(**

**@username VARCHAR(25),**

**@empPassword NVARCHAR(MAX),**

**@comPassword NVARCHAR(MAX),**

**@salt NVARCHAR(512)**

**)**

**AS**

**BEGIN**

**SET NOCOUNT ON;**

**BEGIN TRANSACTION**

**INSERT INTO employee**

**VALUES(@username)**

**DECLARE @employeeID INT**

**IF @@ERROR <> 0**

**BEGIN**

**ROLLBACK TRANSACTION**

**RETURN**

**END**

**ELSE**

**BEGIN**

**COMMIT TRANSACTION**

**SET @employeeID = (SELECT @@IDENTITY)**

**INSERT INTO users**

**VALUES(@employeeID, @empPassword, @comPassword, @salt, 0)**

**END**

**END**

**GO**

**-- \*\*\* Select Records \*\*\***

**--select client**

**CREATE PROC selectClient (**

**@clientID INT**

**)**

**AS**

**BEGIN**

**SET NOCOUNT ON;**

**SELECT \***

**FROM client**

**WHERE clientID = @clientID**

**END**

**GO**

**--select prescription for certain client**

**CREATE PROC (**

**@clientID INT**

**)**

**AS**

**BEGIN**

**SET NOCOUNT ON;**

**SELECT \***

**FROM prescription**

**WHERE clientID = @clientID**

**END**

**GO**

**--select refill by client**

**CREATE PROC selectRefillPerClient (**

**@clientID INT**

**)**

**AS**

**BEGIN**

**SET NOCOUNT ON;**

**SELECT r.refillID, r.prescriptionID, r.dosage, r.frequency,**

**r.supplyDays, r.quantitySupplied, r.amountDue, r.dateOfRefill**

**FROM refill r**

**LEFT JOIN prescription p**

**ON r.prescriptionID = p.prescriptionID**

**WHERE p.clientID = @clientID**

**END**

**GO**

--select client info and prescription info

CREATE PROC selectClientPrescription (

@clientID INT

)

AS

BEGIN

SET NOCOUNT ON;

SELECT p.prescriptionID, p.clientID, c.fName + ' ' + c.lName AS 'Client Name',

p.physicianID, p.medicineID, p.startDate, p.expiryDate, p.refillCounter

FROM prescription AS p

INNER JOIN client AS c on c.clientID = p.clientID

WHERE p.clientID = @clientID

END

GO

**CREATE PROC selectPrescriptionPrice (**

**@prescriptionID INT**

**)**

**AS**

**BEGIN**

**SET NOCOUNT ON;**

**SELECT price**

**FROM prescription**

**WHERE prescriptionID = @prescriptionID**

**IF @@ERROR <> 0**

**THROW 51000, 'There is no prescription with that ID.',1;**

**END**

**GO**

**--select refills**

**CREATE PROC selectRefill (**

**@refillID INT**

**)**

**AS**

**BEGIN**

**SET NOCOUNT ON;**

**SELECT \***

**FROM refill**

**WHERE refillID = @refillID**

**END**

**GO**

**CREATE PROC selectPrescription (**

**@prescriptionID INT**

**)**

**AS**

**BEGIN**

**SET NOCOUNT ON;**

**SELECT \***

**FROM prescription**

**WHERE prescriptionID = @prescriptionID**

**END**

**GO**

-- \*\*\* Update Records \*\*\*

**-- update client**

**CREATE PROC updateClient (**

**@clientID INT,**

**@fName VARCHAR(25),**

**@mInitial CHAR(1) = '',**

**@lName VARCHAR(25),**

**@street1 VARCHAR(100) = '',**

**@street2 VARCHAR(100) = '',**

**@city VARCHAR(25) = '',**

**@addr\_state VARCHAR(25) = '',**

**@zip VARCHAR(10) = '',**

**@phone VARCHAR(15) = '',**

**@email VARCHAR(100) = '',**

**@gender CHAR(1),**

**@dateOfBirth DATE**

**)**

**AS**

**BEGIN**

**SET NOCOUNT ON;**

**BEGIN TRANSACTION**

**UPDATE client**

**SET fName = @fName,**

**mInitial = @mInitial,**

**lName = @lName,**

**street1 = @street1,**

**street2 = @street2,**

**city = @city,**

**addr\_state = @addr\_state,**

**zip = @zip,**

**phone = @phone,**

**email = @email,**

**gender = @gender,**

**dateOfBirth = @dateOfBirth**

**WHERE clientID = @clientID**

**IF @@ERROR <> 0**

**BEGIN**

**ROLLBACK TRANSACTION**

**RAISERROR('Unable to update record.',16,1)**

**RETURN -1**

**END**

**ELSE**

**BEGIN**

**COMMIT TRANSACTION**

**PRINT('Record updated successfully.')**

**END**

**END**

**GO**

**CREATE PROC returnClientIDByUsername(**

**@username VARCHAR(25)**

**)**

**AS**

**BEGIN**

**SET NOCOUNT ON;**

**BEGIN**

**SELECT clientID**

**FROM client**

**WHERE username = @username**

**END**

**END**

**GO**

**CREATE PROC updateClientUserPass (**

**@clientID INT,**

**@newUsername VARCHAR(25),**

**@userPassword NVARCHAR(MAX),**

**@comPassword NVARCHAR(MAX),**

**@salt NVARCHAR(512)**

**)**

**AS**

**BEGIN**

**SET NOCOUNT ON;**

**BEGIN TRANSACTION**

**--update username to the new one**

**UPDATE client**

**SET username = @newUsername**

**WHERE clientID = @clientID**

**--update password to the new one**

**UPDATE users**

**SET userPassword = @userPassword,**

**comPassword = @comPassword,**

**salt = @salt**

**WHERE userID = @clientID**

**IF @@ERROR <> 0**

**BEGIN**

**ROLLBACK TRANSACTION**

**RAISERROR('Unable to update record.',16,1)**

**RETURN -1**

**END**

**ELSE**

**BEGIN**

**COMMIT TRANSACTION**

**PRINT('Record updated successfully')**

**END**

**END**

**GO**

**CREATE PROC updatePrescription (**

**@prescriptionID INT,**

**@clientID INT,**

**@physicianID INT,**

**@medicineID INT,**

**@expiryDate DATE,**

**@refillCounter TINYINT**

**)**

**AS**

**BEGIN**

**SET NOCOUNT ON;**

**BEGIN TRANSACTION**

**IF @clientID != (SELECT clientID FROM prescription WHERE @prescriptionID = prescriptionID)**

**BEGIN**

**UPDATE prescription**

**SET clientID = @clientID**

**WHERE prescriptionID = @prescriptionID**

**END**

**IF @physicianID != (SELECT physicianID FROM prescription WHERE @prescriptionID = prescriptionID)**

**BEGIN**

**UPDATE prescription**

**SET physicianID = @physicianID**

**WHERE prescriptionID = @prescriptionID**

**END**

**IF @medicineID != (SELECT medicineID FROM prescription WHERE @prescriptionID = prescriptionID)**

**BEGIN**

**UPDATE prescription**

**SET medicineID = @clientID**

**WHERE prescriptionID = @prescriptionID**

**END**

**IF @expiryDate != (SELECT expiryDate FROM prescription WHERE @prescriptionID = prescriptionID)**

**BEGIN**

**UPDATE prescription**

**SET expiryDate = @expiryDate**

**WHERE prescriptionID = @prescriptionID**

**END**

**IF @refillCounter != (SELECT refillCounter FROM prescription WHERE @prescriptionID = prescriptionID)**

**BEGIN**

**UPDATE prescription**

**SET refillCounter = @refillCounter**

**WHERE prescriptionID = @prescriptionID**

**END**

**IF @@ERROR <> 0**

**BEGIN**

**ROLLBACK TRANSACTION**

**RAISERROR('Unable to update record.',16,1)**

**RETURN -1**

**END**

**ELSE**

**BEGIN**

**COMMIT TRANSACTION**

**PRINT('Record updated successfully!')**

**END**

**END**

**GO**

**--update refill**

**CREATE PROC updateRefill (**

**@refillID INT,**

**@prescriptionID INT,**

**@dosage VARCHAR(50),**

**@frequency VARCHAR(50),**

**@supplyDays TINYINT,**

**@quantitySupplied TINYINT**

**)**

**AS**

**BEGIN**

**SET NOCOUNT ON;**

**BEGIN TRANSACTION**

**IF @prescriptionID != (SELECT prescriptionID FROM refill WHERE @refillID = refillID)**

**BEGIN**

**UPDATE refill**

**SET prescriptionID = @prescriptionID**

**WHERE refillID = @refillID**

**END**

**IF @dosage != (SELECT dosage FROM refill WHERE @refillID = refillID)**

**BEGIN**

**UPDATE refill**

**SET dosage = @dosage**

**WHERE refillID = @refillID**

**END**

**IF @frequency != (SELECT frequency FROM refill WHERE @refillID = refillID)**

**BEGIN**

**UPDATE refill**

**SET frequency = @frequency**

**WHERE refillID = @refillID**

**END**

**IF @supplyDays != (SELECT supplyDays FROM refill WHERE @refillID = refillID)**

**BEGIN**

**UPDATE refill**

**SET supplyDays = @supplyDays**

**WHERE refillID = @refillID**

**END**

**IF @quantitySupplied != (SELECT quantitySupplied FROM refill WHERE @refillID = refillID)**

**BEGIN**

**UPDATE refill**

**SET quantitySupplied = @quantitySupplied**

**WHERE refillID = @refillID**

**END**

**IF @@ERROR <> 0**

**BEGIN**

**ROLLBACK TRANSACTION**

**RAISERROR('Unable to update record.',16,1)**

**RETURN -1**

**END**

**ELSE**

**BEGIN**

**COMMIT TRANSACTION**

**PRINT('Record updated successfully!')**

**END**

**END**

**GO**

**-- \*\*\* Delete Records \*\*\***

**--delete refill**

**CREATE PROC deleteRefill (**

**@refillID INT**

**)**

**AS**

**BEGIN**

**SET NOCOUNT ON;**

**BEGIN TRANSACTION**

**DELETE**

**FROM payment**

**WHERE refillID = @refillID**

**DELETE**

**FROM refill**

**WHERE refillID = @refillID**

**IF @@ERROR <> 0**

**BEGIN**

**ROLLBACK TRANSACTION**

**RAISERROR('Unable to delete record.',16,1)**

**RETURN -1**

**END**

**ELSE**

**BEGIN**

**COMMIT TRANSACTION**

**PRINT('Record deleted successfully!')**

**END**

**END**

**GO**

**--used for verifying if the login info is correct**

**CREATE PROC getLoginInfo(**

**@username NVARCHAR(64)**

**)**

**AS**

**BEGIN**

**SET NOCOUNT ON;**

**IF EXISTS( SELECT 1 FROM client WHERE username = @username)**

**BEGIN**

**SELECT u.userPassword, u.salt, c.username**

**FROM client c**

**INNER JOIN users u on c.clientID = u.userID**

**WHERE c.username = @username**

**END**

**ELSE**

**BEGIN**

**IF EXISTS( SELECT 1 FROM employee WHERE username = @username)**

**SELECT u.userPassword, u.salt, e.username**

**FROM employee e**

**INNER JOIN users u on e.employeeID = u.userID**

**WHERE e.username = @username**

**ELSE**

**RETURN 0**

**END**

**IF @@ERROR <> 0**

**BEGIN**

**RETURN 3**

**END**

**END**

**GO**

**--returns 0 if username is not found in db**

**--returns 1 if username is a client**

**--returns 2 if username is an employee**

**CREATE PROC findUsername (**

**@username VARCHAR(25)**

**)**

**AS**

**BEGIN**

**SET NOCOUNT ON;**

**IF EXISTS( SELECT 1 FROM client WHERE username = @username)**

**BEGIN**

**RETURN 1;**

**END**

**ELSE**

**BEGIN**

**IF EXISTS( SELECT 1 FROM employee WHERE username = @username)**

**RETURN 2;**

**ELSE**

**RETURN 0;**

**END**

**END**

**GO**

-- \*\*\* TRIGGERS \*\*\*

--check refill counter and expiry date when creating new refill

CREATE TRIGGER trg\_checkRefillCounter

ON refill

FOR INSERT

AS

BEGIN

DECLARE @prescriptionID INT

DECLARE @refillCount TINYINT

DECLARE @expiryDate DATE

SELECT @prescriptionID = (SELECT prescriptionID FROM inserted)

SELECT @refillCount = (SELECT refillCounter

FROM prescription

WHERE prescriptionID = @prescriptionID)

SELECT @expiryDate = (SELECT expiryDate

FROM prescription

WHERE prescriptionID = @prescriptionID)

IF @refillCount = 0 OR @expiryDate < GETDATE()

THROW 51000, 'This prescription has expired/reached the refill limit.', 1;

ELSE

BEGIN

UPDATE prescription

SET refillCounter = @refillCount - 1

WHERE prescriptionID = @prescriptionID

END

IF @@ERROR <> 0

RAISERROR('Unable to insert record.',16,1)

END

GO

--trigger for price of prescription

CREATE TRIGGER trg\_setPrescriptionPrice

ON prescription

FOR INSERT

AS

BEGIN

DECLARE @medicineID INT

DECLARE @medicineCost DECIMAL(7,2)

DECLARE @clientID INT

DECLARE @insuranceID INT

DECLARE @insuranceCoverage DECIMAL(5,2)

DECLARE @prescriptionID INT

DECLARE @prescriptionPrice DECIMAL(7,2)

SELECT @medicineID = (SELECT medicineID FROM inserted)

SELECT @medicineCost = (SELECT cost

FROM medicine

WHERE medicineID = @medicineID)

SELECT @clientID = (SELECT clientID FROM inserted)

SELECT @insuranceID = (SELECT insuranceID

FROM clientInsurance

WHERE clientID = @clientID)

SELECT @insuranceCoverage = (SELECT coveragePercent

FROM insurance

WHERE insuranceID = @insuranceID)

SELECT @prescriptionID = (SELECT prescriptionID FROM inserted)

SET @prescriptionPrice = @medicineCost - (@medicineCost / @insuranceCoverage)

UPDATE prescription

SET price = @prescriptionPrice

WHERE prescriptionID = @prescriptionID

IF @@ERROR <> 0

BEGIN

RAISERROR('Unable to insert record.',16,1)

END

END

GO

--set refill price

CREATE TRIGGER trg\_setRefillPrice

ON refill

FOR INSERT

AS

BEGIN

DECLARE @prescriptionID INT

DECLARE @refillID INT

DECLARE @prescriptionPrice DECIMAL(7,2)

SELECT @prescriptionID = (SELECT prescriptionID FROM inserted)

SELECT @refillID = (SELECT refillID FROM inserted)

SELECT @prescriptionPrice = (SELECT price

FROM prescription

WHERE prescriptionID = @prescriptionID)

UPDATE refill

SET amountDue = @prescriptionPrice

WHERE refillID = @refillID

IF @@ERROR <> 0

BEGIN

RAISERROR('Unable to insert record.',16,1)

END

END

GO

--audit log: refill

CREATE TRIGGER trg\_updateRefillAudit

ON refill

AFTER UPDATE

AS

BEGIN

DECLARE @tableName VARCHAR(50) = 'refill'

DECLARE @fieldName VARCHAR(50)

DECLARE @recordID INT

DECLARE @oldValue VARCHAR(255)

DECLARE @newValue VARCHAR(255)

DECLARE @changeDate DATE = GETDATE()

IF UPDATE(prescriptionID)

BEGIN

SET @fieldName = 'prescriptionID'

SELECT @recordID = (SELECT refillID FROM inserted)

SELECT @oldValue = (SELECT prescriptionID FROM deleted)

SELECT @newValue = (SELECT prescriptionID FROM inserted)

INSERT INTO auditLog

VALUES(@tableName, @fieldName, @recordID, @oldValue, @newValue, @changeDate)

END

IF UPDATE(dosage)

BEGIN

SET @fieldName = 'dosage'

SELECT @recordID = (SELECT refillID FROM inserted)

SELECT @oldValue = (SELECT dosage FROM deleted)

SELECT @newValue = (SELECT dosage FROM inserted)

INSERT INTO auditLog

VALUES(@tableName, @fieldName, @recordID, @oldValue, @newValue, @changeDate)

END

IF UPDATE(frequency)

BEGIN

SET @fieldName = 'frequency'

SELECT @recordID = (SELECT refillID FROM inserted)

SELECT @oldValue = (SELECT frequency FROM deleted)

SELECT @newValue = (SELECT frequency FROM inserted)

INSERT INTO auditLog

VALUES(@tableName, @fieldName, @recordID, @oldValue, @newValue, @changeDate)

END

IF UPDATE(supplyDays)

BEGIN

SET @fieldName = 'supplyDays'

SELECT @recordID = (SELECT refillID FROM inserted)

SELECT @oldValue = (SELECT supplyDays FROM deleted)

SELECT @newValue = (SELECT supplyDays FROM inserted)

INSERT INTO auditLog

VALUES(@tableName, @fieldName, @recordID, @oldValue, @newValue, @changeDate)

END

IF UPDATE(quantitySupplied)

BEGIN

SET @fieldName = 'quantitySupplied'

SELECT @recordID = (SELECT refillID FROM inserted)

SELECT @oldValue = (SELECT quantitySupplied FROM deleted)

SELECT @newValue = (SELECT quantitySupplied FROM inserted)

INSERT INTO auditLog

VALUES(@tableName, @fieldName, @recordID, @oldValue, @newValue, @changeDate)

END

IF UPDATE(amountDue)

BEGIN

SET @fieldName = 'amountDue'

SELECT @recordID = (SELECT refillID FROM inserted)

SELECT @oldValue = (SELECT amountDue FROM deleted)

SELECT @newValue = (SELECT amountDue FROM inserted)

INSERT INTO auditLog

VALUES(@tableName, @fieldName, @recordID, @oldValue, @newValue, @changeDate)

END

IF UPDATE(dateOfRefill)

BEGIN

SET @fieldName = 'dateOfRefill'

SELECT @recordID = (SELECT refillID FROM inserted)

SELECT @oldValue = (SELECT dateOfRefill FROM deleted)

SELECT @newValue = (SELECT dateOfRefill FROM inserted)

INSERT INTO auditLog

VALUES(@tableName, @fieldName, @recordID, @oldValue, @newValue, @changeDate)

END

IF @@ERROR <> 0

BEGIN

RAISERROR('Unable to insert record.',16,1)

END

END

GO

--audit log: prescription

CREATE TRIGGER trg\_updatePrescriptionAudit

ON prescription

AFTER UPDATE

AS

BEGIN

DECLARE @tableName VARCHAR(50) = 'prescription'

DECLARE @fieldName VARCHAR(50)

DECLARE @recordID INT

DECLARE @oldValue VARCHAR(255)

DECLARE @newValue VARCHAR(255)

DECLARE @changeDate DATE = GETDATE()

IF UPDATE(clientID)

BEGIN

SET @fieldName = 'clientID'

SELECT @recordID = (SELECT prescriptionID FROM inserted)

SELECT @oldValue = (SELECT clientID FROM deleted)

SELECT @newValue = (SELECT clientID FROM inserted)

INSERT INTO auditLog

VALUES(@tableName, @fieldName, @recordID, @oldValue, @newValue, @changeDate)

END

IF UPDATE(physicianID)

BEGIN

SET @fieldName = 'physicianID'

SELECT @recordID = (SELECT prescriptionID FROM inserted)

SELECT @oldValue = (SELECT physicianID FROM deleted)

SELECT @newValue = (SELECT physicianID FROM inserted)

INSERT INTO auditLog

VALUES(@tableName, @fieldName, @recordID, @oldValue, @newValue, @changeDate)

END

IF UPDATE(medicineID)

BEGIN

SET @fieldName = 'medicineID'

SELECT @recordID = (SELECT prescriptionID FROM inserted)

SELECT @oldValue = (SELECT medicineID FROM deleted)

SELECT @newValue = (SELECT medicineID FROM inserted)

INSERT INTO auditLog

VALUES(@tableName, @fieldName, @recordID, @oldValue, @newValue, @changeDate)

END

IF UPDATE(startDate)

BEGIN

SET @fieldName = 'startDate'

SELECT @recordID = (SELECT prescriptionID FROM inserted)

SELECT @oldValue = (SELECT startDate FROM deleted)

SELECT @newValue = (SELECT startDate FROM inserted)

INSERT INTO auditLog

VALUES(@tableName, @fieldName, @recordID, @oldValue, @newValue, @changeDate)

END

IF UPDATE(expiryDate)

BEGIN

SET @fieldName = 'expiryDate'

SELECT @recordID = (SELECT prescriptionID FROM inserted)

SELECT @oldValue = (SELECT expiryDate FROM deleted)

SELECT @newValue = (SELECT expiryDate FROM inserted)

INSERT INTO auditLog

VALUES(@tableName, @fieldName, @recordID, @oldValue, @newValue, @changeDate)

END

IF UPDATE(refillCounter)

BEGIN

SET @fieldName = 'refillCounter'

SELECT @recordID = (SELECT prescriptionID FROM inserted)

SELECT @oldValue = (SELECT refillCounter FROM deleted)

SELECT @newValue = (SELECT refillCounter FROM inserted)

INSERT INTO auditLog

VALUES(@tableName, @fieldName, @recordID, @oldValue, @newValue, @changeDate)

END

IF UPDATE(price)

BEGIN

SET @fieldName = 'price'

SELECT @recordID = (SELECT prescriptionID FROM inserted)

SELECT @oldValue = (SELECT price FROM deleted)

SELECT @newValue = (SELECT price FROM inserted)

INSERT INTO auditLog

VALUES(@tableName, @fieldName, @recordID, @oldValue, @newValue, @changeDate)

END

IF @@ERROR <> 0

BEGIN

RAISERROR('Unable to insert record.',16,1)

END

END

GO

--audit log: payment

CREATE TRIGGER trg\_updatePaymentAudit

ON payment

AFTER UPDATE

AS

BEGIN

DECLARE @tableName VARCHAR(50) = 'payment'

DECLARE @fieldName VARCHAR(50)

DECLARE @recordID INT

DECLARE @oldValue VARCHAR(255)

DECLARE @newValue VARCHAR(255)

DECLARE @changeDate DATE = GETDATE()

IF UPDATE(refillID)

BEGIN

SET @fieldName = 'refillID'

SELECT @recordID = (SELECT paymentID FROM inserted)

SELECT @oldValue = (SELECT refillID FROM deleted)

SELECT @newValue = (SELECT refillID FROM inserted)

INSERT INTO auditLog

VALUES(@tableName, @fieldName, @recordID, @oldValue, @newValue, @changeDate)

END

IF UPDATE(paymentType)

BEGIN

SET @fieldName = 'paymentType'

SELECT @recordID = (SELECT paymentID FROM inserted)

SELECT @oldValue = (SELECT paymentType FROM deleted)

SELECT @newValue = (SELECT paymentType FROM inserted)

INSERT INTO auditLog

VALUES(@tableName, @fieldName, @recordID, @oldValue, @newValue, @changeDate)

END

IF UPDATE(dateOfPayment)

BEGIN

SET @fieldName = 'dateOfPayment'

SELECT @recordID = (SELECT paymentID FROM inserted)

SELECT @oldValue = (SELECT dateOfPayment FROM deleted)

SELECT @newValue = (SELECT dateOfPayment FROM inserted)

INSERT INTO auditLog

VALUES(@tableName, @fieldName, @recordID, @oldValue, @newValue, @changeDate)

END

IF @@ERROR <> 0

BEGIN

RAISERROR('Unable to insert record.',16,1)

END

END