**IST 278 Unit 6 Lab**

Name: Ramon Rodriguez Date: 6/10/24

# **Lab Instructions**

1. Type your name and the date in the spaces provided.
2. Use the SQL Server Management Studio and the IST278EagleCorp database.
3. Complete all five exercises in this lab
4. Upload and submit a completed copy of this lab sheet before the due date.

**Transaction Lab**

In this lab you will code and test a stored procedure that uses transaction logic to do an explicit BEGIN TRAN, COMMIT TRAN, and ROLLBACK TRAN.

# **1. Unit 6 Lab Exercise 1**

Use the IST278EagleCorp database and write a stored procedure named SPInsertSalesLeadsXX where the XX are your initials. This stored procedure is to receive twelve optional parameters, do limited validation, and use the twelve parameters to insert a row into the SalesLeadsXX table. Before you get started with the coding of your SPInsertSalesLeadsXX stored procedure verify that you have the SalesLeadsXX table created in your copy of the IST278EagleCorp database. If you do not have it, refer to UNIT 5 lab exercise 6 and create that table now.

Once you have verified the existence of your SalesLeadsXX table begin coding your SPInsertSalesLeadsXX stored procedure per the following directions:

1. Code the CREATE PROC SPInsertSalesLeadsXX (where the XX are your initials) line of code followed by the 12 parameters. Make sure to give each of the twelve parameters a default value of NULL so that they will be optional. For example, the @LeadID parameter and the @CustFirstName parameter should be declared as follows:

@LeadID varchar(10) = NULL,

@CustFirstName varchar(15) =NULL

1. Code the word AS on the line following the 12 parameters
2. Code logic to validate the existence of non-NULL values in the @LeadID, @CustFirstName, @CustLastName, and @Phone since they are required attributes on the SalesLeadsXX table. For example, for the @LeadID and @CustFirstName your validation code should be similar to the following:

-- validating that a value was provided for @LeadID

IF @LeadID IS NULL

BEGIN

PRINT 'You must enter a LeadID'

PRINT ' Lead not inserted.'

PRINT ' '

RETURN

END

-- LeadID must be begin with L or Z (extra validation thrown for practice)

IF (SUBSTRING(@LeadID, 1, 1) not in ('L','Z'))

BEGIN

PRINT 'Invalid LeadID ''' + @LeadID + ''

PRINT ' LeadID must begin with either a ''L'' or ''Z''.'

PRINT ' Customer not inserted.'

PRINT ' '

RETURN

END

-- validating that a value was provided for @CustFirstName

IF @CustFirstName IS NULL

BEGIN

PRINT 'You must enter a customer first name'

PRINT ' Lead not inserted.'

PRINT ' '

RETURN

END

1. After the validation logic code BEGIN TRAN;
2. After the BEGIN TRAN code an insert statement that uses the twelve parameters to insert a row in the SalesLeadsXX table. Hint: if you want to see an example of an insert statement in a stored procedure using parameters, there is an example on page 471 of an insert to the Invoices table that is using a few parameters.
3. After the Insert Statement code logic to check to see if the insert should be committed or rolled back and do which ever is appropriate. For example, this logic should be similar to the following:

IF EXISTS (SELECT \* FROM Customer WHERE CustFirstName = @CustFirstName

AND CustLastName = @CustLastName AND CompanyName = @CompanyName)

BEGIN

PRINT 'Lead ' + @CompanyName + ' already exists in Customer table.'

PRINT ' Lead not inserted in the SalesLeadsXX table.'

PRINT ' '

ROLLBACK TRAN

END

ELSE

BEGIN

COMMIT TRAN

PRINT 'Lead Inserted '+ @LeadID

PRINT ' '

END

Note: Do this validation after the insert as sketched out above. It would be more efficient to avoid the ROLLBACK by coding this validation before the insert along with the other data validation that you coded, but in exercise we want to practice using the ROLLBACK and COMMIT so we are doing this last bit of validation after the insert.

1. Code a comment after the above logic that states it is the end of your stored procedure and include your name in it. Example follows

-- END OF STORED PROCEDURE Coded by Bob Whaite

**---** **Paste below this line the Stored Procedure you wrote for this exercise --**

Paste here

DROP PROCEDURE IF EXISTS SPInsertSalesLeadsRR

GO

CREATE PROCEDURE SPInsertSalesLeadsRR

@LeadID NVARCHAR(10) = NULL,

@CompanyName NVARCHAR(100) = NULL,

@CustFirstName NVARCHAR(50) = NULL,

@CustLastName NVARCHAR(50) = NULL,

@CustTitle NVARCHAR(50) = NULL,

@Address NVARCHAR(200) = NULL,

@City NVARCHAR(100) = NULL,

@State NVARCHAR(50) = NULL,

@PostalCode NVARCHAR(20) = NULL,

@Phone NVARCHAR(20) = NULL,

@Fax NVARCHAR(20) = NULL,

@EmailAddr NVARCHAR(100) = NULL

AS

IF @LeadID IS NULL

BEGIN

PRINT 'You must enter a LeadID'

PRINT 'Lead not inserted'

PRINT ''

RETURN

END

IF (SUBSTRING(@LeadID, 1, 1) not in ('L','Z'))

BEGIN

PRINT 'Invalid LeadID ''' + @LeadID + ''

PRINT ' LeadID must begin with either a ''L'' or ''Z''.'

PRINT ' Customer not inserted.'

PRINT ' '

RETURN

END

IF @CustFirstName IS NULL

BEGIN

PRINT 'You must enter a customer first name'

PRINT 'Lead not inserted'

PRINT ''

RETURN

END

IF @CustLastName IS NULL

BEGIN

PRINT 'You must enter a customer last name'

PRINT 'Lead not inserted'

PRINT ''

RETURN

END

IF @Phone IS NULL

BEGIN

PRINT 'You must enter a phone number'

PRINT 'Lead not inserted'

PRINT ''

RETURN

END

BEGIN TRAN

INSERT INTO SalesLeadsRR (LeadID, CompanyName, CustFirstName, CustLastName, CustTitle, Address,

City, State, PostalCode, Phone, Fax, EmailAddr)

VALUES( @LeadID, @CompanyName, @CustFirstName, @CustLastName, @CustTitle, @Address,

@City, @State, @PostalCode, @Phone, @Fax, @EmailAddr);

IF EXISTS (SELECT \* FROM Customer WHERE CustFirstName = @CustFirstName

AND CustLastName = @CustLastName AND CompanyName = @CompanyName)

BEGIN

PRINT 'Lead ' + @CompanyName + ' already exists in Customer table.'

PRINT ' Lead not inserted in the SalesLeadsXX table.'

PRINT ' '

ROLLBACK TRAN

END

ELSE

BEGIN

COMMIT TRAN

PRINT 'Lead Inserted '+ @LeadID

PRINT ' '

END

-- This is the end of my stored procedure Coded by RAMON RODRIGUEZ

# **2. Unit 6 Lab Exercise 2**

Code a call to the procedure you created in exercise 1 passing values to the parameters as specified below (make sure to use your real name for the custFirstName and CustLastName):

**Column Name Value to pass**

LeadID L-0000011

CompanyName Real Auto

CustFirstName **Your FirstName**

CustLastName **Your LastName**

CustTitle Nothing (let it default to NULL)

Address 666 Cliff Drive

City Pickens

State SC

PostalCode 22222

Phone 864 222-4444

Fax Nothing (let it default to NULL)

EmailAddr Nothing (let it default to NULL)

**--- Paste below this line the call statement you wrote for exercise 2 –**

Paste here

EXEC SPInsertSalesLeadsRR @LeadID = 'L-0000011', @CompanyName = 'Real Auto', @CustFirstName = 'Ramon', @CustLastName = 'Rodriguez', @CustTitle = NULL, @Address = '666 Cliff Drive', @City = 'Pickens', @State = 'SC', @PostalCode = '22222', @Phone = '864 222-4444', @Fax = NULL, @EmailAddr = NULL;

**---** **Paste below this line the completion status messages from executing the exercise 2 call statement-**

Paste here

A screenshot of a computer

Description automatically generated

# **3. Unit 6 Lab Exercise 3**

Code a call to the procedure you created in exercise 1 passing values to the parameters as specified below:

**Column Name Value to pass**

LeadID L-0000012

CompanyName Fish Tank Cleaning

CustFirstName John

CustLastName Mavrick

CustTitle Nothing (let it default to NULL)

Address 777 Pond Drive

City Pickens

State SC

PostalCode 22222

Phone Nothing

Fax Nothing (let it default to NULL)

EmailAddr Nothing (let it default to NULL)

**--- Paste below this line the call statement you wrote for exercise 3 –**

Paste here

EXEC SPInsertSalesLeadsRR @LeadID = 'L-0000012', @CompanyName = 'Fish Tank Cleaning', @CustFirstName = 'John', @CustLastName = 'Mavrick', @CustTitle = NULL, @Address = '777 Pond Drive', @City = 'Pickens', @State = 'SC', @PostalCode = '22222', @Phone = NULL, @Fax = NULL, @EmailAddr = NULL;

**---** **Paste below this line the completion status messages from executing the exercise 3 call statement-**

Paste here

A white background with black text

Description automatically generated

# **4. Unit 6 Lab Exercise 4**

Code a call to the procedure you created in exercise 1 passing values to the parameters as specified below:

**Column Name Value to pass**

LeadID L-0000013

CompanyName Dixon Pharmacy

CustFirstName Tracy

CustLastName Cicholski

CustTitle Nothing (let it default to NULL)

Address 1920 Albion St.

City Crystal Springs

State MS

PostalCode 39059

Phone 601-959-1315

Fax Nothing (let it default to NULL)

EmailAddr Nothing (let it default to NULL)

**--- Paste below this line the call statement you wrote for exercise 4 –**

Paste here

EXEC SPInsertSalesLeadsRR @LeadID = 'L-0000013', @CompanyName = 'Dixon Pharmacy', @CustFirstName = 'Tracy', @CustLastName = 'Cicholski', @CustTitle = NULL, @Address = '1920 Albion St.', @City = 'Crystal Springs', @State = 'MS', @PostalCode = '39059', @Phone = '601-959-1315', @Fax = NULL, @EmailAddr = NULL;

**---** **Paste below this line the completion status messages from executing the exercise 4 call statement-**

Paste here

A close-up of a message

Description automatically generated

# **5. Unit 6 Lab Exercise 5**

Execute a SELECT \* FROM your SalesLeadsXX table and Paste the result set below:

**---** **Paste below this line the result set from running a SELECT \* FROM your SalesLeadsXX table-**

Paste here

A screenshot of a computer

Description automatically generated