# Murach Chapter 15 How to code Stored Procedures Assignments

## Assignment 1: Creating a Procedure with INPUT & OUTPUT parameters.

Create a procedure that uses a cursor to generate the below report, which contains the first and last name of the customer that bought a Hrd. cloth, 1/4-in., 2x50. Use the tables: sales\_co.CUSTOMER, sales\_co.product, sales\_co.line, and sales\_co.invoice i

use db2;

go

declare

@first1 nvarchar(30),

@last1 nvarchar(30),

@product1 nvarchar(30);

set @product1 = 'Hrd. cloth, 1/4-in., 2x50';

exec proc1 @first1 output, @last1 output, @product1;

print '=============================================';

print @first1 + ' ' + @last1 + ' Bought ' + @product1;

print '=============================================';

Format you output as show on example document.

Answer:

create proc proc1 (

@first1 nvarchar(30) output,

@last1 nvarchar(30) output,

@product1 nvarchar(30)

)

as

DECLARE nameCursor CURSOR for

SELECT CUS\_LNAME,CUS\_FNAME,P\_DESCRIPT

from sales\_co.invoice i

join sales\_co.CUSTOMER c on c.CUS\_CODE= i.CUS\_CODE

join sales\_co.LINE l on l.INV\_decimal=i.INV\_decimal

JOIN sales\_co.PRODUCT p on p.P\_CODE= l.P\_CODE

WHERE P\_DESCRIPT= @product1

open nameCursor

FETCH next from nameCursor INTO @first1, @last1, @product1

WHILE @@FETCH\_STATUS= 0

BEGIN

SET @first1= @first1

SET @last1= @last1

FETCH next from nameCursor INTO @first1, @last1, @product1

END

close nameCursor

DEALLOCATE nameCursor;

Graphical user interface, text, application, email

Description automatically generated

## Assignment 2: Creating a Procedure Removes Rows Of A Table

Create a procedure that removes your name from the table db2.college.students. If your name is not in the database use 'Amber' 'Howard'. If the your name or 'Amber' 'Howard' is not in the database, print out the name + ' is not in the database anymore'

use db2;

go

declare

@last1 nvarchar(30);

set @last1 = 'Gabor';

exec proc2 @last1 ;

Answer:

create PROCEDURE proc2

(

@last1 nvarchar(30)

)

AS

BEGIN

IF EXISTS (SELECT \* FROM [College].[students] WHERE lastname = @last1) --true

BEGIN

SET @last1 = 'Amber'

IF EXISTS (SELECT \* FROM [College].[students] WHERE lastname = @last1) -- false

BEGIN

PRINT @last1 + ' is in the database anymore.'

END

ELSE

BEGIN

PRINT @last1 + ' is not in the database anymore.'

END

END

ELSE

BEGIN

DELETE FROM College.students WHERE lastname = @last1

PRINT @last1 + ' has been deleted';

END

END

Graphical user interface, text, application

Description automatically generated

## Assignment 3: Procedure

Use the db2 database and the MM schema. Create 2 procedures named MOVIE\_RENTAL & MOVIE\_RETURN. One procedure movie\_rental has 4 parameters, 2 input (passed to procedure) and two output (passed out of the procedure). This procedure will take in the title of the movie and the last name of the person renting the movie. It will then populate the output variables with the movie\_id and the member\_id. The second procedure movie\_return has 3 parameters, 2 input one output. movie\_return will be executed by the script in the assignment receiving the title of the movie and the last name of the person renting the movie, calling the procedure movie\_rental at the beginning of the procedure. Using the output of the procedure move\_rental it will determine if the member has rented the movie, if they have rented the movie print out that the member has returned the movie and insert today's date for the check\_in date.

See examples of solution.

use movies\_db

go

declare

@message varchar(200);

begin

exec dbo.movie\_return 'Maulder', 'The Fifth Element', @message output;

print @message;

END;

select \* from mm.movie;

select \* from mm.rental;

Answer:

create proc movieReturn (@title varchar(50), @last varchar(50), @message varchar(200) output)

AS

BEGIN

DECLARE @movie\_id INT,@member\_id INT, @date nvarchar(30)

SELECT @date= CONVERT(varchar(10), GETDATE(), 111)

EXEC movie\_rental @last, @title, @movie\_id output, @member\_id output

IF LEN(@member\_id) >0

BEGIN

SET @message= @last + 'has returned the movie. last check in date' + @date

END

ELSE

BEGIN

SET @message= @last + 'has no rented movie title.' + @title

END

END

create proc movie\_rental (@title varchar(50), @last varchar(50), @movie\_id INT output,@member\_id INT output )

AS

BEGIN

SELECT @movie\_id =rental.movie\_id, @member\_id=rental.member\_id from mm.rental

JOIN mm.movie on movie.movie\_title= @title

JOIN mm.member on member.last = @last

END

Table

Description automatically generated

Table

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated