--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*--

-- Title: Assignment08

-- Author: AAlem

-- Desc: This file demonstrates how to use Stored Procedures

-- Change Log: When,Who,What

-- 2022-08-31, AAlem,Completed File

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*--

Begin Try

Use Master;

If Exists(Select Name From SysDatabases Where Name = 'Assignment08DB\_AAlem')

Begin

Alter Database [Assignment08DB\_AAlem] set Single\_user With Rollback Immediate;

Drop Database Assignment08DB\_AAlem;

End

Create Database Assignment08DB\_AAlem;

End Try

Begin Catch

Print Error\_Number();

End Catch

go

Use Assignment08DB\_AAlem;

-- Create Tables (Module 01)--

Create Table Categories

([CategoryID] [int] IDENTITY(1,1) NOT NULL

,[CategoryName] [nvarchar](100) NOT NULL

);

go

Create Table Products

([ProductID] [int] IDENTITY(1,1) NOT NULL

,[ProductName] [nvarchar](100) NOT NULL

,[CategoryID] [int] NULL

,[UnitPrice] [money] NOT NULL

);

go

Create Table Employees -- New Table

([EmployeeID] [int] IDENTITY(1,1) NOT NULL

,[EmployeeFirstName] [nvarchar](100) NOT NULL

,[EmployeeLastName] [nvarchar](100) NOT NULL

,[ManagerID] [int] NULL

);

go

Create Table Inventories

([InventoryID] [int] IDENTITY(1,1) NOT NULL

,[InventoryDate] [Date] NOT NULL

,[EmployeeID] [int] NOT NULL -- New Column

,[ProductID] [int] NOT NULL

,[Count] [int] NOT NULL

);

go

-- Add Constraints (Module 02) --

Begin -- Categories

Alter Table Categories

Add Constraint pkCategories

Primary Key (CategoryId);

Alter Table Categories

Add Constraint ukCategories

Unique (CategoryName);

End

go

Begin -- Products

Alter Table Products

Add Constraint pkProducts

Primary Key (ProductId);

Alter Table Products

Add Constraint ukProducts

Unique (ProductName);

Alter Table Products

Add Constraint fkProductsToCategories

Foreign Key (CategoryId) References Categories(CategoryId);

Alter Table Products

Add Constraint ckProductUnitPriceZeroOrHigher

Check (UnitPrice >= 0);

End

go

Begin -- Employees

Alter Table Employees

Add Constraint pkEmployees

Primary Key (EmployeeId);

Alter Table Employees

Add Constraint fkEmployeesToEmployeesManager

Foreign Key (ManagerId) References Employees(EmployeeId);

End

go

Begin -- Inventories

Alter Table Inventories

Add Constraint pkInventories

Primary Key (InventoryId);

Alter Table Inventories

Add Constraint dfInventoryDate

Default GetDate() For InventoryDate;

Alter Table Inventories

Add Constraint fkInventoriesToProducts

Foreign Key (ProductId) References Products(ProductId);

Alter Table Inventories

Add Constraint ckInventoryCountZeroOrHigher

Check ([Count] >= 0);

Alter Table Inventories

Add Constraint fkInventoriesToEmployees

Foreign Key (EmployeeId) References Employees(EmployeeId);

End

go

-- Adding Data (Module 04) --

-- NOTE: We are starting without data this time!

-- Adding Views (Module 06) --

Create View vCategories With SchemaBinding

AS

Select CategoryID, CategoryName From dbo.Categories;

go

Create View vProducts With SchemaBinding

AS

Select ProductID, ProductName, CategoryID, UnitPrice From dbo.Products;

go

Create View vEmployees With SchemaBinding

AS

Select EmployeeID, EmployeeFirstName, EmployeeLastName, ManagerID From dbo.Employees;

go

Create View vInventories With SchemaBinding

AS

Select InventoryID, InventoryDate, EmployeeID, ProductID, [Count] From dbo.Inventories;

go

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Questions and Answers \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* NOTE:Use the following template to create your stored procedures and plan on this taking ~2-3 hours

Create Procedure <pTinsCategories

(@CategoryName nvarch(100)) =

-- Author:

-- Desc: Processes <Desc text>

-- Change Log: When,Who,What

-- <2022-08-31>,<Your Name Here>,Completed Sproc.

AS

Begin

Declare @RC int = 0;

Begin Try

Begin Transaction

-- Transaction Code --

Commit Transaction

Set @RC = +1

End Try

Begin Catch

Rollback Transaction

Print Error\_Message()

Set @RC = -1

End Catch

Return @RC;

End

go

\*/

-- Question 1 (20 pts): How can you create Insert, Update, and Delete Transactions Store Procedures

-- for the Categories table?

--Create Procedure pInsCategories

Create Procedure pInsCategories

(@CategoryName nvarchar(100))

-- Author: <AAlem>

-- Desc: Processes <Inserts into Categories table>

-- Change Log: When,Who,What

-- <2022-08-31>,<AAlem>>,Completed Sproc.

AS

Begin

Declare @RC int = 0;

Begin Try

Begin Transaction

Insert into Categories

(CategoryName)

Values(@CategoryName)

Commit Transaction

Set @RC = +1

End Try

Begin Catch

Rollback Transaction

Print Error\_Message()

Set @RC = -1

End Catch

Return @RC;

End

go

-----------------------------------------------------------

---Exec pInsCategories @CategoryName='CatB';

-----Select \* from Categories order by 1

------------------------------------------------------------

--Create Procedure pUpdCategories

Create Procedure pUpdCategories

(@CategoryID int, @CategoryName nvarchar(100))

-- Author: <AAlem>

-- Desc: Processes <Updates Data in Categories table>

-- Change Log: When,Who,What

-- <2022-08-31>,<AAlem>>,Completed Sproc.

AS

Begin

Declare @RC int = 0;

Begin Try

Begin Transaction

Update Categories

Set CategoryName=@CategoryName

Where CategoryID=@CategoryID

Commit Transaction

Set @RC = +1

End Try

Begin Catch

Rollback Transaction

Print Error\_Message()

Set @RC = -1

End Catch

Return @RC;

End

go

--Create Procedure pDelCategories

--< Place Your Code Here!>--

Create Procedure pDelCategories

(@CategoryID int)

-- Author: <AAlem>

-- Desc: Processes <Deletes Data from Categories table>

-- Change Log: When,Who,What

-- <2022-08-31>,<AAlem>>,Completed Sproc.

AS

Begin

Declare @RC int = 0;

Begin Try

Begin Transaction

Delete Categories

Where CategoryID=@CategoryID

Commit Transaction

Set @RC = +1

End Try

Begin Catch

Rollback Transaction

Print Error\_Message()

Set @RC = -1

End Catch

Return @RC;

End

go

---Select \* from Categories

-- Question 2 (20 pts): How can you create Insert, Update, and Delete Transactions Store Procedures

-- for the Products table?

--Create Procedure pInsProducts

--< Place Your Code Here!>--

Create Procedure pInsProducts

(@ProductName nvarchar(100),@CategoryID int, @UnitPrice money)

-- Author: <AAlem>

-- Desc: Processes <Inserts into Products table>

-- Change Log: When,Who,What

-- <2022-08-31>,<AAlem>>,Completed Sproc.

AS

Begin

Declare @RC int = 0;

Begin Try

Begin Transaction

Insert into Products

(ProductName, CategoryID, UnitPrice )

Values (@ProductName,@CategoryID, @UnitPrice )

Commit Transaction

Set @RC = +1

End Try

Begin Catch

Rollback Transaction

Print Error\_Message()

Set @RC = -1

End Catch

Return @RC;

End

go

-----------------

--Create Procedure pUpdProducts

--< Place Your Code Here!>--

Create Procedure pUpdProducts

(@ProductName nvarchar(100),@CategoryID int, @UnitPrice money)

-- Author: <AAlem>

-- Desc: Processes <Updates Data in Products table>

-- Change Log: When,Who,What

-- <2022-08-31>,<AAlem>>,Completed Sproc.

AS

Begin

Declare @RC int = 0;

Begin Try

Begin Transaction

Update Products

Set ProductName=@ProductName, UnitPrice=@UnitPrice

Where CategoryID= @CategoryID

Commit Transaction

Set @RC = +1

End Try

Begin Catch

Rollback Transaction

Print Error\_Message()

Set @RC = -1

End Catch

Return @RC;

End

go

--Create Procedure pDelProducts

--< Place Your Code Here!>--

Create Procedure pDelProducts

(@ProductName nvarchar(100),@CategoryID int, @UnitPrice money)

-- Author: <AAlem>

-- Desc: Processes <Deletes Data in Products table>

-- Change Log: When,Who,What

-- <2022-08-31>,<AAlem>>,Completed Sproc.

AS

Begin

Declare @RC int = 0;

Begin Try

Begin Transaction

Delete Products

Where CategoryID=@CategoryID

Commit Transaction

Set @RC = +1

End Try

Begin Catch

Rollback Transaction

Print Error\_Message()

Set @RC = -1

End Catch

Return @RC;

End

go

-- Question 3 (20 pts): How can you create Insert, Update, and Delete Transactions Store Procedures

-- for the Employees table?

--Create Procedure pInsEmployees

--< Place Your Code Here!>--

Create Procedure pInsEmployees

(@EmployeeFirstName nvarchar(100), @EmployeeLastName nvarchar(100),@ManagerID int)

-- Author: <AAlem>

-- Desc: Processes <Inserts into Employees table>

-- Change Log: When,Who,What

-- <2022-08-31>,<AAlem>>,Completed Sproc.

AS

Begin

Declare @RC int = 0;

Begin Try

Begin Transaction

Insert into Employees

(EmployeeFirstName,EmployeeLastName,ManagerID)

Values ( @EmployeeFirstName,@EmployeeLastName,@ManagerID )

Commit Transaction

Set @RC = +1

End Try

Begin Catch

Rollback Transaction

Print Error\_Message()

Set @RC = -1

End Catch

Return @RC;

End

go

--Create Procedure pUpdEmployees

--< Place Your Code Here!>--

Create Procedure pUpdEmployees

(@EmployeeFirstName nvarchar(100), @EmployeeLastName nvarchar(100),@ManagerID int)

-- Author: <AAlem>

-- Desc: Processes <Updates date in Employees table>

-- Change Log: When,Who,What

-- <2022-08-31>,<AAlem>>,Completed Sproc.

AS

Begin

Declare @RC int = 0;

Begin Try

Begin Transaction

Update Employees

Set EmployeeFirstName= @EmployeeFirstName,EmployeeLastName= @EmployeeLastName, ManagerID=@ManagerID

Where ManagerID= @ManagerID

Commit Transaction

Set @RC = +1

End Try

Begin Catch

Rollback Transaction

Print Error\_Message()

Set @RC = -1

End Catch

Return @RC;

End

go

--Create Procedure pDelEmployees

--< Place Your Code Here!>--

Create Procedure pDelEmployees

(@EmployeeFirstName nvarchar(100), @EmployeeLastName nvarchar(100),@ManagerID int)

-- Author: <AAlem>

-- Desc: Processes <Deletes in Employees table>

-- Change Log: When,Who,What

-- <2022-08-31>,<AAlem>>,Completed Sproc.

AS

Begin

Declare @RC int = 0;

Begin Try

Begin Transaction

Delete Employees

Where ManagerID= @ManagerID

Commit Transaction

Set @RC = +1

End Try

Begin Catch

Rollback Transaction

Print Error\_Message()

Set @RC = -1

End Catch

Return @RC;

End

go

-- Question 4 (20 pts): How can you create Insert, Update, and Delete Transactions Store Procedures

-- for the Inventories table?

--Create Procedure pInsInventories

--< Place Your Code Here!>--

Create Procedure pInsInventories

(@InventoryDate Date, @EmployeeID int,@ProductID int,@Count nvarchar(100))

-- Author: <AAlem>

-- Desc: Processes <Inserts into Inventoreis table>

-- Change Log: When,Who,What

-- <2022-08-31>,<AAlem>>,Completed Sproc.

AS

Begin

Declare @RC int = 0;

Begin Try

Begin Transaction

Insert into Inventories

(InventoryDate, EmployeeID,ProductID,Count )

Values ( @InventoryDate, @EmployeeID,@ProductID,@Count )

Commit Transaction

Set @RC = +1

End Try

Begin Catch

Rollback Transaction

Print Error\_Message()

Set @RC = -1

End Catch

Return @RC;

End

go

--Create Procedure pUpdInventories

--< Place Your Code Here!>--

Create Procedure pUpdInventories

(@InventoryDate Date, @EmployeeID int,@ProductID int,@Count nvarchar(100))

-- Author: <AAlem>

-- Desc: Processes <Updates Data in Inventoreis table>

-- Change Log: When,Who,What

-- <2022-08-31>,<AAlem>>,Completed Sproc.

AS

Begin

Declare @RC int = 0;

Begin Try

Begin Transaction

Update Inventories

Set InventoryDate=@InventoryDate, EmployeeID=@EmployeeID,ProductID=@ProductID,

Count=@Count

Where ProductID=@ProductID

Commit Transaction

Set @RC = +1

End Try

Begin Catch

Rollback Transaction

Print Error\_Message()

Set @RC = -1

End Catch

Return @RC;

End

go

--Create Procedure pDelInventories

--< Place Your Code Here!>--

Create Procedure pDelInventories

(@InventoryDate Date, @EmployeeID int,@ProductID int,@Count nvarchar(100))

-- Author: <AAlem>

-- Desc: Processes <Deletes Data in Inventoreis table>

-- Change Log: When,Who,What

-- <2022-08-31>,<AAlem>>,Completed Sproc.

AS

Begin

Declare @RC int = 0;

Begin Try

Begin Transaction

Delete Inventories

Where ProductID=@ProductID

Commit Transaction

Set @RC = +1

End Try

Begin Catch

Rollback Transaction

Print Error\_Message()

Set @RC = -1

End Catch

Return @RC;

End

go

-- Question 5 (20 pts): How can you Execute each of your Insert, Update, and Delete stored procedures?

-- Include custom messages to indicate the status of each sproc's execution.

-- Here is template to help you get started:

/\*

Declare @Status int;

Exec @Status = <SprocName>

@ParameterName = 'A'

Select Case @Status

When +1 Then '<TableName> Insert was successful!'

When -1 Then '<TableName> Insert failed! Common Issues: Duplicate Data'

End as [Status];

Select \* From <ViewName> Where ColID = 1;

go

\*/

--< Test Insert Sprocs >--

-- Test [dbo].[pInsCategories]

Declare @Status int;

Exec @Status = PInsCategories

@CategoryName = 'CatA'

Select Case @Status

When +1 Then '<Categories> Insert was successful!'

When -1 Then '<Categories> Insert failed! Common Issues: Duplicate Data'

End as [Status];

Select \* From vCategories Where CategoryID= 1;

go

-- Test [dbo].[pInsProducts]

Declare @Status int;

Exec @Status = pInsProducts

@ProductName = 'ProductNameA'

Select Case @Status

When +1 Then '<Products> Insert was successful!'

When -1 Then '<Products> Insert failed! Common Issues: Duplicate Data'

End as [Status];

Select \* From vProducts Where ProductID= 1;

go

-- Test [dbo].[pInsEmployees]

Declare @Status int;

Exec @Status = pInsEmployees

@EmployeeID = 'EmployeeID'

Select Case @Status

When +1 Then '<Employees> Insert was successful!'

When -1 Then '<Employees> Insert failed! Common Issues: Duplicate Data'

End as [Status];

Select \* From vEmployees Where EmployeeID= 1;

go

-- Test [dbo].[pInsInventories]

Declare @Status int;

Declare @Status int;

Exec @Status = pInsInventories

@InventoryID = 'InventoryID'

Select Case @Status

When +1 Then '<Inventories Insert was successful!'

When -1 Then '<Inventories> Insert failed! Common Issues: Duplicate Data'

End as [Status];

Select \* From vInventories Where InventoryID= 1;

go

--< Test Update Sprocs >--

-- Test Update [dbo].[pUpdCategories]

Declare @Status int;

Exec @Status = pUpdCategories

@CategoryName = 'CatA'

Select Case @Status

When +1 Then '<Categories Insert was successful!'

When -1 Then '<Categories> Insert failed! Common Issues: Duplicate Data'

End as [Status];

Select \* From vCategories Where CategoryID= 1;

go

-- Test [dbo].[pUpdProducts]

Declare @Status int;

Exec @Status = pUpdProducts

@ProductName = 'ProductNameA'

Select Case @Status

When +1 Then '<Products> Insert was successful!'

When -1 Then '<Products> Insert failed! Common Issues: Duplicate Data'

End as [Status];

Select \* From vProducts Where ProductID= 1;

go

-- Test [dbo].[pUpdEmployees]

Declare @Status int;

Exec @Status = pUpdEmployees

@EmployeeID = 'EmployeeID'

Select Case @Status

When +1 Then '<Employees> Insert was successful!'

When -1 Then '<Employees> Insert failed! Common Issues: Duplicate Data'

End as [Status];

Select \* From vEmployees Where EmployeeID= 1;

go

-- Test [dbo].[pUpdInventories]

Declare @Status int;

Declare @Status int;

Exec @Status = pUpdInventories

@InventoryID = 'InventoryID'

Select Case @Status

When +1 Then '<Inventories Insert was successful!'

When -1 Then '<Inventories> Insert failed! Common Issues: Duplicate Data'

End as [Status];

Select \* From vInventories Where InventoryID= 1;

go

--< Test Delete Sprocs >--

-- Test [dbo].[pDelInventories]

Declare @Status int;

Declare @Status int;

Exec @Status = pDelInventories

@InventoryID = 'InventoryID'

Select Case @Status

When +1 Then '<Inventories Insert was successful!'

When -1 Then '<Inventories> Insert failed! Common Issues: Duplicate Data'

End as [Status];

Select \* From vInventories Where InventoryID= 1;

go

-- Test [dbo].[pDelEmployees]

Declare @Status int;

Exec @Status = pDelEmployees

@EmployeeID = 'EmployeeID'

Select Case @Status

When +1 Then '<Employees> Insert was successful!'

When -1 Then '<Employees> Insert failed! Common Issues: Duplicate Data'

End as [Status];

Select \* From vEmployees Where EmployeeID= 1;

go

-- Test [dbo].[pDelProducts]

Declare @Status int;

Exec @Status =pDelProducts

@ProductName = 'ProductNameA'

Select Case @Status

When +1 Then '<Products> Insert was successful!'

When -1 Then '<Products> Insert failed! Common Issues: Duplicate Data'

End as [Status];

Select \* From vProducts Where ProductID= 1;

go

-- Test [dbo].[pDelCategories]

Declare @Status int;

Exec @Status =pDelCategories

@CategoryName = 'CatA'

Select Case @Status

When +1 Then '<Categories Insert was successful!'

When -1 Then '<Categories> Insert failed! Common Issues: Duplicate Data'

End as [Status];

Select \* From vCategories Where CategoryID= 1;

go

--{ IMPORTANT!!! }--

-- To get full credit, your script must run without having to highlight individual statements!!!

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/