use AdventureWorks

go

--Exercise 1

--Scenario:The Production.Product table includes a column called 'ListPrice'. Whenever an update is made to the table ,

-- if either the existing balance or the new balance is greater than 1'000 US Dollars,an entry must be written

-- to the Production.ProductAudit audit table

--Supporting Documentation---

--The Production.ProductAudit table is used to hold changes to high value products.The date to be inserted in each column

--is shown in the following table

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

Column | Data Type | Value to insert

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

AuditID | INT | IDENTITY

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ProductID | INT | ProductID

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

UpdateTime | datetime2 | SYSDATETIME()

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ModifyingUser | varchar(30) | ORIGINAL\_LOGIN()

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

OriginalListPrice |decimal(18,2) | ListPrice before update

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

NewListPrice |decimal(18,2) | ListPrice after update

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

--Task 1

--In SQL Server Management Studio, review the existing structure of the Production.ProductAudit table and the values

-- required in each column , based on the supporting documentation

--Review the existing structure of the Production.Product table on SSMS

go

create table Production.ProductAudit

(AuditID INT IDENTITY(1,1) ,

ProductID INT not null,

UpdateTime datetime2 not null,

ModifyingUser varchar(30) not null,

OriginalListPrice decimal(18,2) not null,

NewListPrice decimal(18,2) not null,

Primary key (AuditID),

Foreign key (ProductID) REFERENCES Production.Product(ProductID)

)

GO

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

Commands completed successfully.

Completion time: 2024-09-11T21:28:13.2873839+02:00

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

--Task2

--Design a trigger

--Design and create a trigger that meets the needs of the supporting documentation.

--T1--

go

create Trigger Production.NewInsertProduct On Production.Product

After Update

as

Begin

Update Production.Product

Set ListPrice = d.ListPrice,ModifiedDate = getdate()

From inserted d inner join Production.Product p On d.ProductID = p.ProductID

Where p.ProductID IN

(Select I.ProductID From INSERTED as I inner join inserted d ON I.ProductID = d.ProductID

Where I.ListPrice > 1000 or d.ListPrice = 1000)

End

go

--T2--

alter trigger Production.newUserUpdateProduct On Production.Product

After update

as

begin

INSERT INTO Production.ProductAudit(ProductID,UpdateTime,ModifyingUser,OriginalListPrice,NewListPrice)

Select i.ProductID ,SYSDATETIME(),ORIGINAL\_LOGIN(),d.ListPrice,i.ListPrice

From inserted i inner join deleted as d ON i.ProductID = d.ProductID

Where d.ListPrice = 1000 or i.ListPrice > 1000

END

GO

--Task3

--Test the Behavior of the trigger

--Execute data modification statements that are designated to test whether the trigger is working as expected

--Për T1-----

Update p

Set p.ListPrice = 1600

From Production.Product as p

Where p.ProductID = 5

--Për T2-----

update p

Set ListPrice = 2006

From Production.Product as p

Where p.ProductID between 1 and 999

Select \* From Production.ProductAudit

--Exercise 2

--Scenario

--Now that the trigger created in the first exercise has been deployed to production,the operations team

--is complaining that too many entries are being audited. Many accounts have more than 10'000 US dollars as

-- a balance and minor movements of money are causing audit entries.

--1--You must modify the trigger so that 'only changes in the balance' of more than 10'000 US dollars are

-- audited instead

--The main tasks for this exercise are:

--Task 1:Modify the trigger

--1--Review the design of the existing trigger and design what modifications are required

go

create schema Marketing

go

Commands completed successfully.

Completion time: 2024-09-12T13:18:28.8876084+02:00

go

Create table Marketing.AccountBalance

(AccountId int IDENTITY(1,1) not null,

ModifiedDate datetime2 not null,

CurrentBalance money not null,

Primary key (AccountId)

)

Commands completed successfully.

Completion time: 2024-09-12T13:37:31.0007993+02:00

create table Marketing.CampaignAudit

(AuditId int identity(1,1) not null,

AccountID int not null,

ModifyingUser nvarchar(50) not null,

UpdatedTime datetime2 not null,

CurrentBalanceMoney money not null,

UpdatedBalanceMoney money not null,

PRIMARY KEY(AuditId),

Foreign key(AccountID) REFERENCES Marketing.AccountBalance(AccountId)

)

Commands completed successfully.

Completion time: 2024-09-12T13:37:59.0804974+02:00

--2--Use an ALTER TRIGGER statement to change the existing trigger so that it will meet the updated requirements

go

alter trigger Marketing.TR\_CampaignBalance\_Update

on Marketing.AccountBalance

after update

as

begin

Insert into Marketing.CampaignAudit(AccountID,ModifyingUser,UpdatedTime,CurrentBalanceMoney,UpdatedBalanceMoney)

Select i.AccountId ,ORIGINAL\_LOGIN(),SYSDATETIME(),d.CurrentBalance,i.CurrentBalance

From inserted i inner join deleted d ON i.AccountId = d.AccountId

Where abs(d.CurrentBalance - i.CurrentBalance) > 10000 And d.CurrentBalance > 10000

end

go

Commands completed successfully.

Completion time: 2024-09-12T13:55:54.4036102+02:00

--Task 2:Delete all rows from Marketing.CampaignAudit table

DELETE FROM Marketing.CampaignAudit

go

--Execute a DELETE statement to remove all existing rows from the Marketing.CampaignAudit table

(0 rows affected)

Completion time: 2024-09-12T13:56:48.8442400+02:00

--Task 3:Test the modified trigger

--1-- Execute data modification statements that are designated to test whether the trigger is working as expected

create procedure insertMarketingBalance(@Balance money)

as

begin

Insert into Marketing.AccountBalance(ModifiedDate,CurrentBalance)

Values(getDate(),@Balance)

end

go

Commands completed successfully.

Completion time: 2024-09-12T14:05:38.7102510+02:00

Exec insertMarketingBalance @Balance = 66000

go

(1 row affected)

Completion time: 2024-09-12T14:05:57.2696696+02:00

Select \* From Marketing.AccountBalance

go

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

AccountId | ModifiedDate | CurrentBalance

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

1 | 2024-09-12 14:05:57.2530000 | 60000.00

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

go

alter proc MoveMarketingBalance(@AccountId int OUTPUT,@Balance money)

as

begin

UPDATE ab

SET ab.CurrentBalance = @Balance + ab.CurrentBalance

From Marketing.AccountBalance as ab

Where ab.AccountId = @AccountId

end

go

Commands completed successfully.

Completion time: 2024-09-12T14:15:23.7168299+02:00

Declare @BankAccountId int =6,@UpdatedBalance money = -20000 --terheqje

Exec MoveMarketingBalance @AccountId = @BankAccountId OUTPUT,@Balance = @UpdatedBalance

Select @BankAccountId as BankAccountId , @UpdatedBalance as moveBalanced

go

| BankAccountId | moveBalanced

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

1 | 1 | - 2000.00

Select \* From Marketing.AccountBalance

AccountId| ModifiedDate | CurrentBalance

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

1 | 2024-09-12 14:05:57.2530000| -4000.00

--2--Close SQL SERVER Management Studio without saving anything

Select \* From Marketing.CampaignAudit