Sri Lanka Institute of Information

Technology

**Database Management Systems**

**for**

**Securit**

**y**

**-**

**IE2024**

Lab Submission 0

5

**IT22151056**

**De Silva K.R.K.D**

**Activity 01**

# 01)

CREATE VIEW TopThreeCustomers AS

SELECT TOP 3

C.custName AS CustomerName,

B.branchName AS BranchName,

A.balance AS Balance,

B.b\_address AS BranchAddress

FROM

Customer1 C

JOIN Belongs\_to BT ON C.custNo = BT.custNo

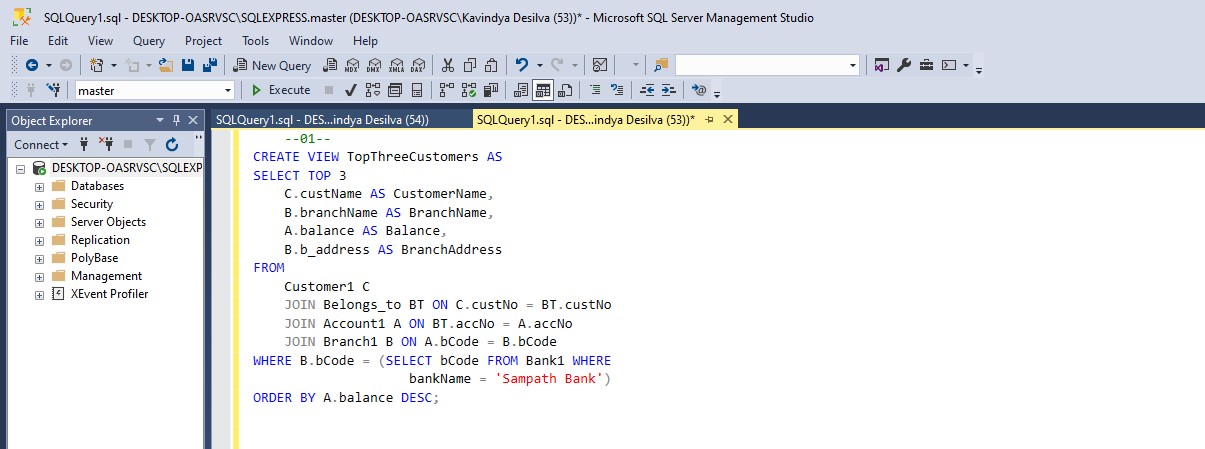
JOIN Account1 A ON BT.accNo = A.accNo

JOIN Branch1 B ON A.bCode = B.bCode

WHERE B.bCode = (SELECT bCode FROM Bank1 WHERE

bankName = 'Sampath Bank')

ORDER BY A.balance DESC;



# 02)

CREATE VIEW TopDepositByBank AS

SELECT

B.bankName AS BankName,

COUNT (H.tid) AS TotalDeposits

FROM

Bank1 B

JOIN Branch1 BR ON B.bCode = BR.bCode

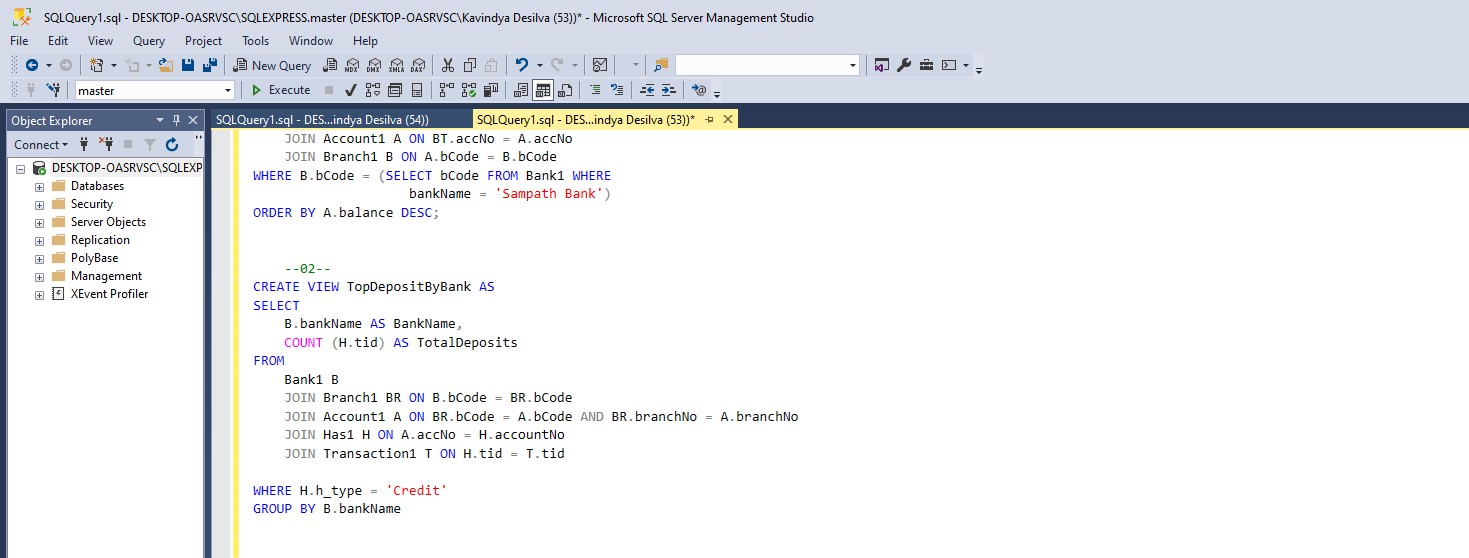
JOIN Account1 A ON BR.bCode = A.bCode AND BR.branchNo = A.branchNo

JOIN Has1 H ON A.accNo = H.accountNo

JOIN Transaction1 T ON H.tid = T.tid

WHERE H.h\_type = 'Credit'

GROUP BY B.bankName



# 03)

CREATE FUNCTION GetTotalBranchBalance (

@bankName varchar (200),

@branchName varchar (150)

)

RETURNS DECIMAL(10,2)

AS

BEGIN

DECLARE @totalBalance DECIMAL (10,2)

SELECT @totalBalance = SUM(A.balance)

FROM Bank1 B

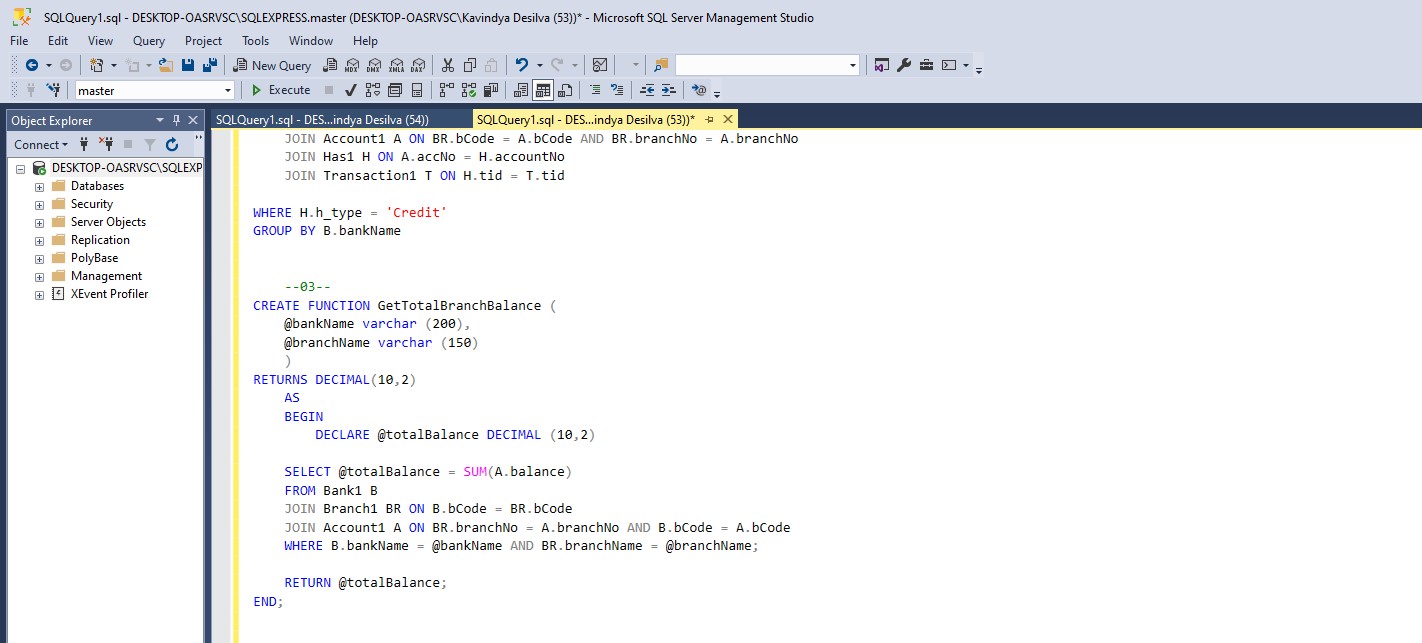
JOIN Branch1 BR ON B.bCode = BR.bCode

JOIN Account1 A ON BR.branchNo = A.branchNo AND B.bCode = A.bCode

WHERE B.bankName = @bankName AND BR.branchName = @branchName;

RETURN @totalBalance;

END;



# 04)

CREATE FUNCTION GetTotWithdrawalsByCustomer (

@cusNo int,

@year int,

@method varchar(100)

)

RETURNS DECIMAL(10, 2)

AS

BEGIN

DECLARE @totalWithdrawal DECIMAL(10,2);

SELECT @totalWithdrawal = SUM(T.amount)

FROM Transaction1 T

JOIN Has1 H ON T.tid = H.tid

JOIN Account A ON H.accountNo = A.accNo

JOIN Belongs\_to BE ON A.accNo = BE.accNo

WHERE BE.cuNo = @cusNo

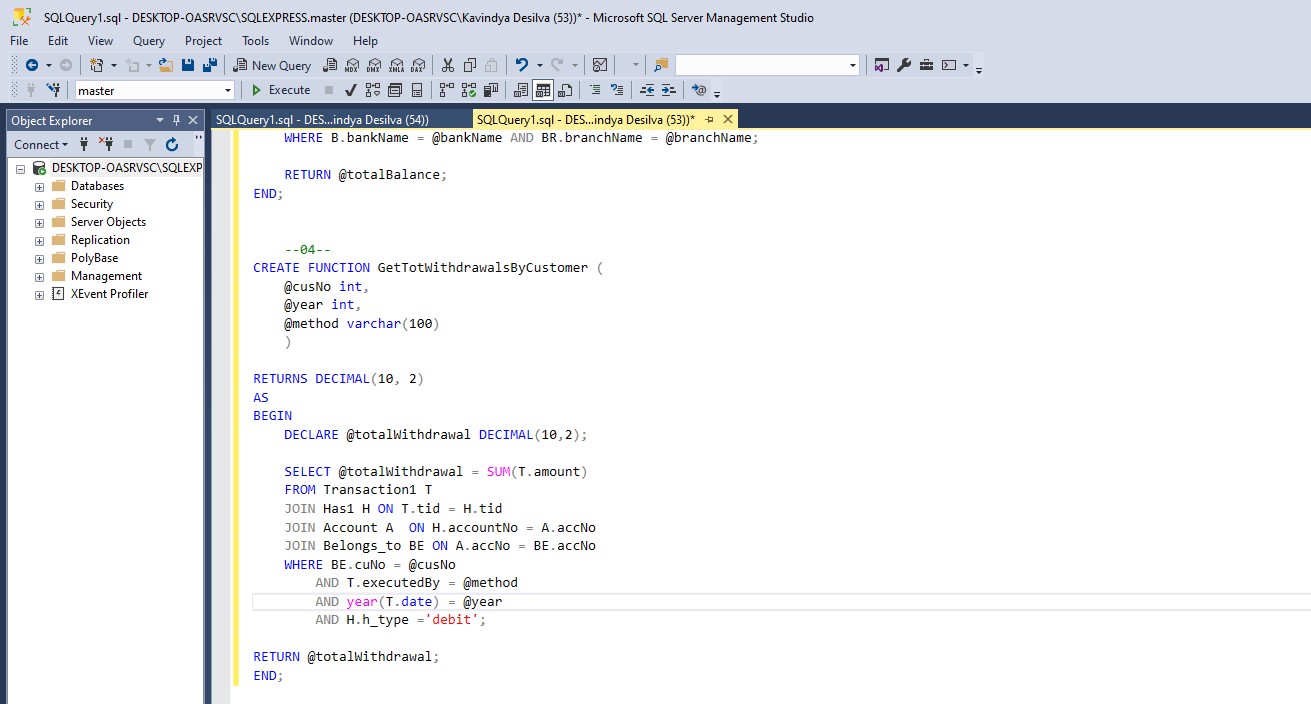
AND T.executedBy = @method

AND year(T.date) = @year

AND H.h\_type ='debit';

RETURN @totalWithdrawal;

END;



# 05)

CREATE PROCEDURE UpdateAccBalance

@accNo int,

@operation varchar(20),

@amt DECIMAL(10,2)

AS

BEGIN

IF @operation NOT IN ('Credit', 'Debit')

BEGIN

RAISEERROR('Invalid operation. Operation must be either "Credit" or "Debit."', 16, 1);

RETURN;

END;

IF @operation ='Credit'

BEGIN

UPDATE Account1

SET balance = balance + @amt

WHERE accNo = @accNo;

END;

ELSE IF @operation ='Debit'

BEGIN

DECLARE @currentBalance DECIMAL(10,2);

SELECT @currentBalance =balance

FROM Account1

WHERE accNo = @accNo

IF @amt > @currentBalance

BEGIN

RAISEERROR('insufficient balance for the debit', 16,1);

RETURN;

END;

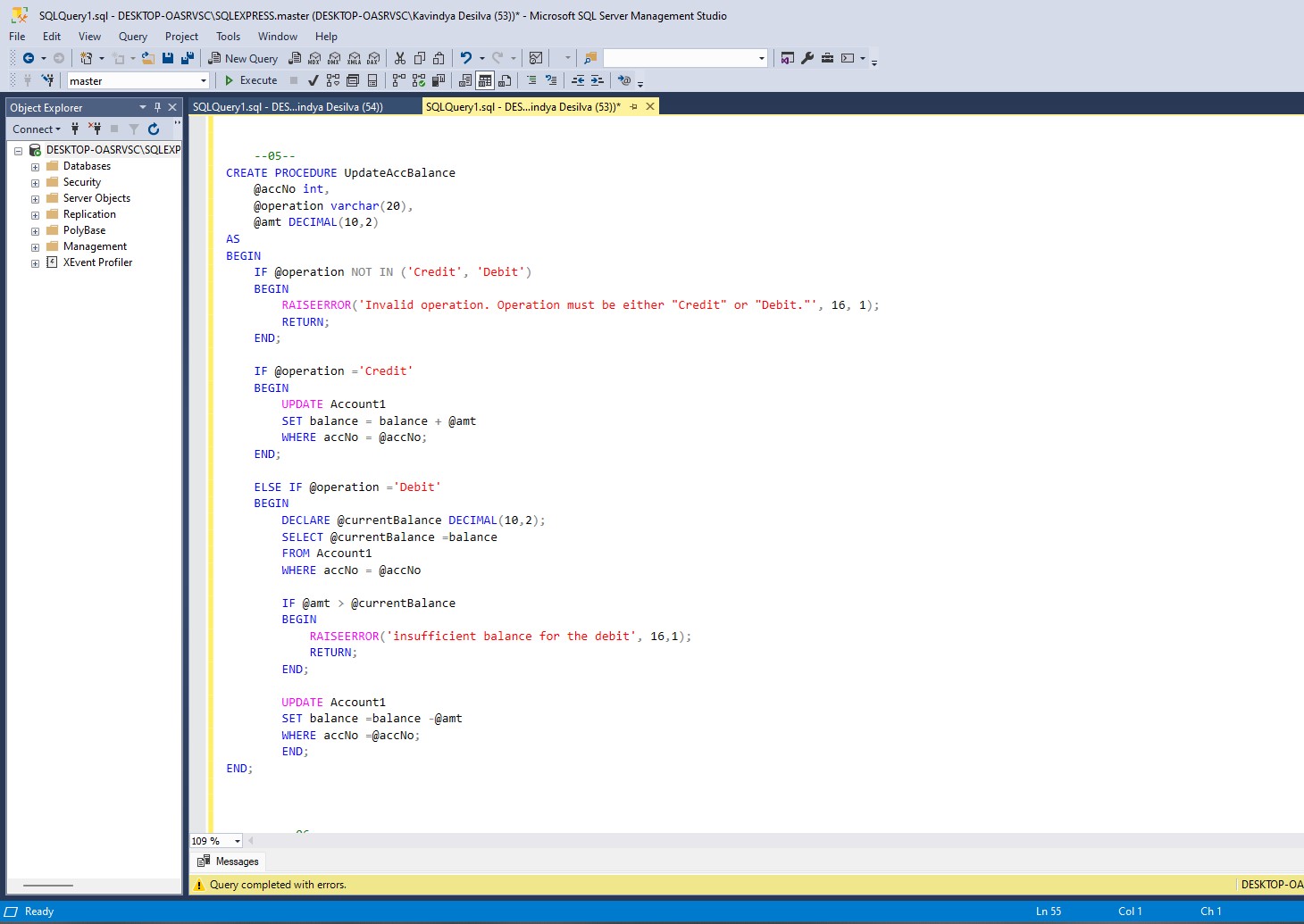
UPDATE Account1

SET balance =balance -@amt

WHERE accNo =@accNo;

END;

END;



# 06)

CREATE PROCEDURE TransMoney

@sourceAccNo int,

@amt DECIMAL(10,2),

@targetAccNo int

AS

BEGIN

IF @sourceAccNo = @targetAccNo

BEGIN

RAISEERROR('Target and source accounts cannot be the same.', 16, 1); RETURN;

END;

DECLARE @sourceBalance DECIMAL(10,2);

SELECT @sourceBalance = balance

FROM Account1

WHERE accNo =@sourceAccNo;

IF @amt > @sourceBalance

BEGIN

RAISEERROR('Amount in the source account is insufficient for the transfer.', 16, 1);

RETURN;

END;

BEGIN TRY

BEGIN Transaction1;

UPDATE Account1

SET balance = balance -@amt

WHERE accNo =@sourceAccNo;

UPDATE Account1

SET balance = balance + @amt

WHERE accNo =@targetAccNo;

COMMIT Transaction1;

END TRY

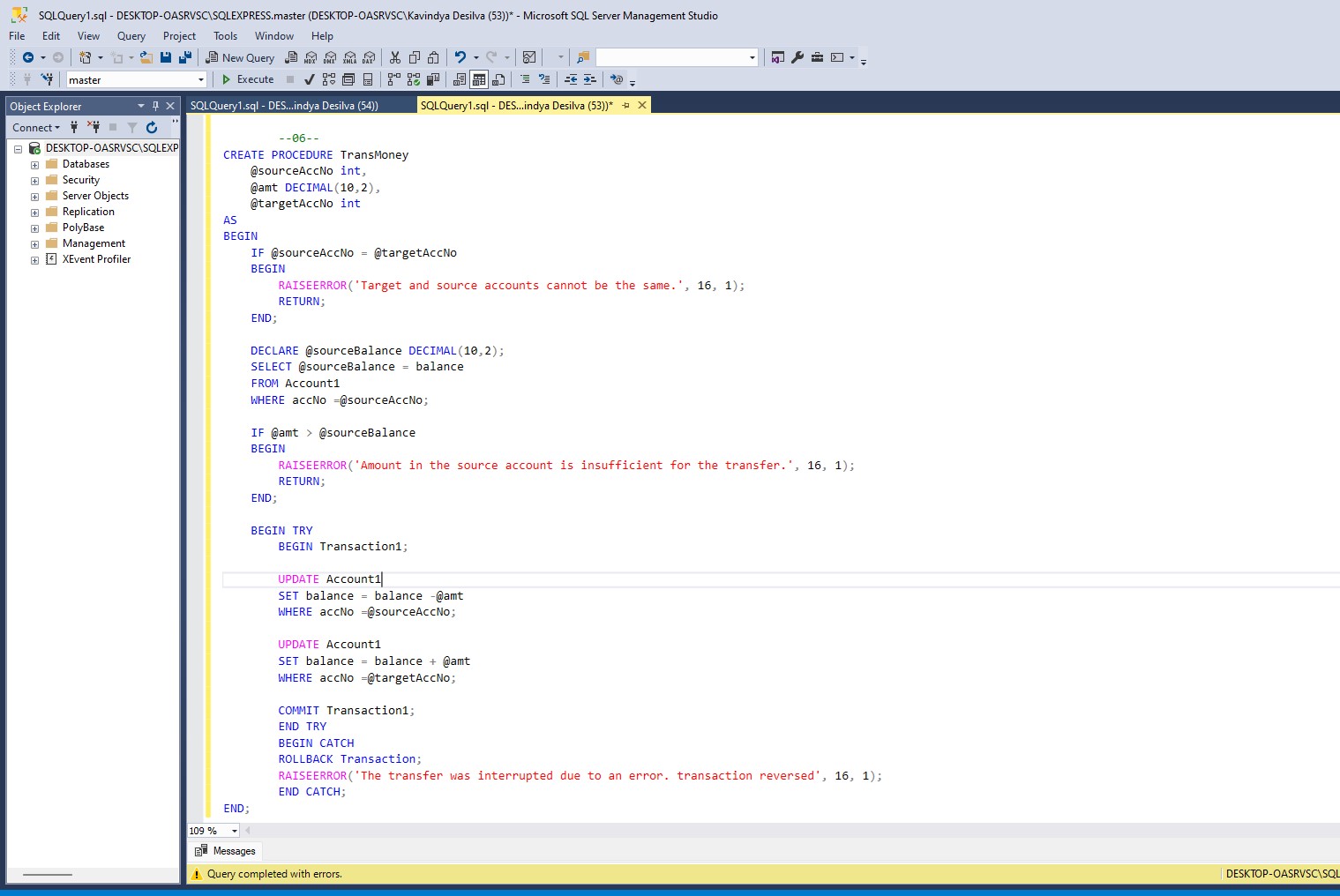
BEGIN CATCH

ROLLBACK Transaction;

RAISEERROR('The transfer was interrupted due to an error. transaction reversed', 16, 1);

END CATCH;

END;



# 07)

CREATE TRIGGER CheckminBalance

ON Account1

AFTER UPDATE

AS BEGIN

IF EXISTS (

SELECT 1

FROM deleted d

JOIN inserted i ON d.accNo = i.accNo

WHERE i.balance < 500

)

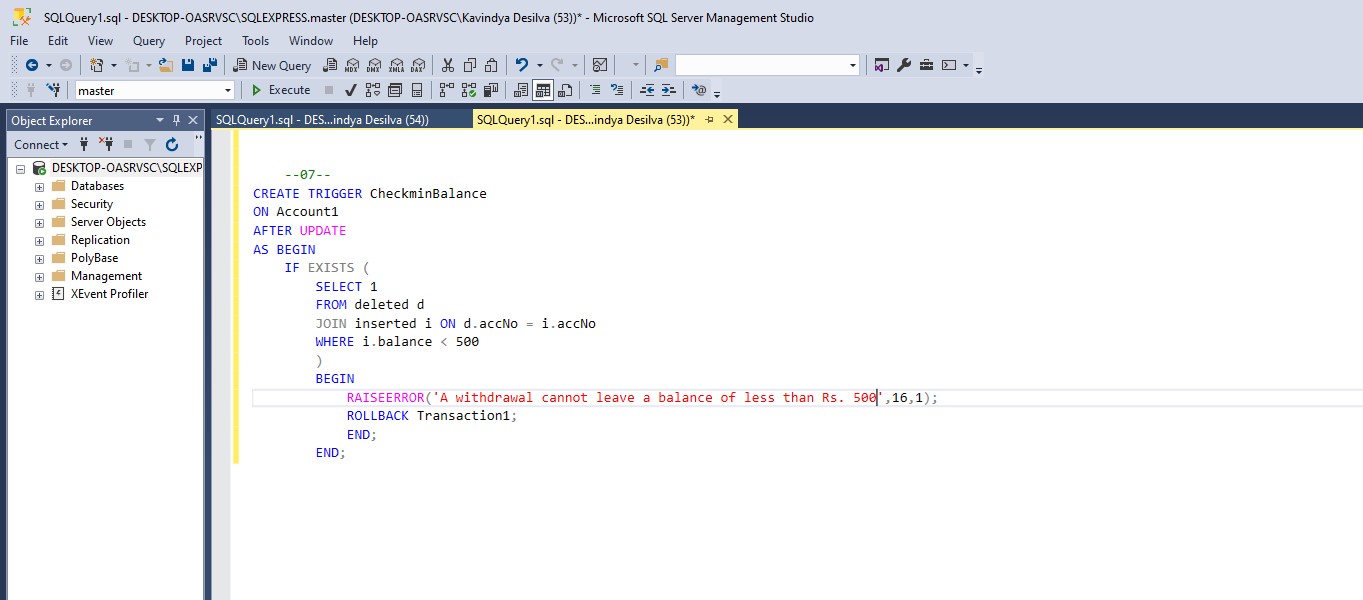
BEGIN

RAISEERROR('A withdrawal cannot leave a balance of less than Rs.

500',16,1);

ROLLBACK Transaction1;

END; END;



# 08)

CREATE TRIGGER CheckDailyAMTWithdrawal

ON Transaction1

AFTER INSERT

AS BEGIN

DECLARE @MaxDailyWithdrawal DECIMAL(10,2) = 80000;

DECLARE @accNo int;

DECLARE @Withdrawalamt DECIMAL(10,2);

DECLARE @WithdrawalDate DATE;

SELECT @accNo = i.accNo, @Withdrawalamt = i.amt, @WithdrawalDate = CONVERT(DATE, i.date)

FROM inserted i;

DECLARE @TotalWithdrawal DECIMAL(10,2);

SELECT @TotalWithdrawal =SUM(amt)

FROM Transaction1

WHERE executeddBy ='ATM'

AND accNo = @AccNo

AND COnvert(DATE, date) = @WithdrawalDate;

IF @TotalWithdrawal > @MaxDailyWithdrawal

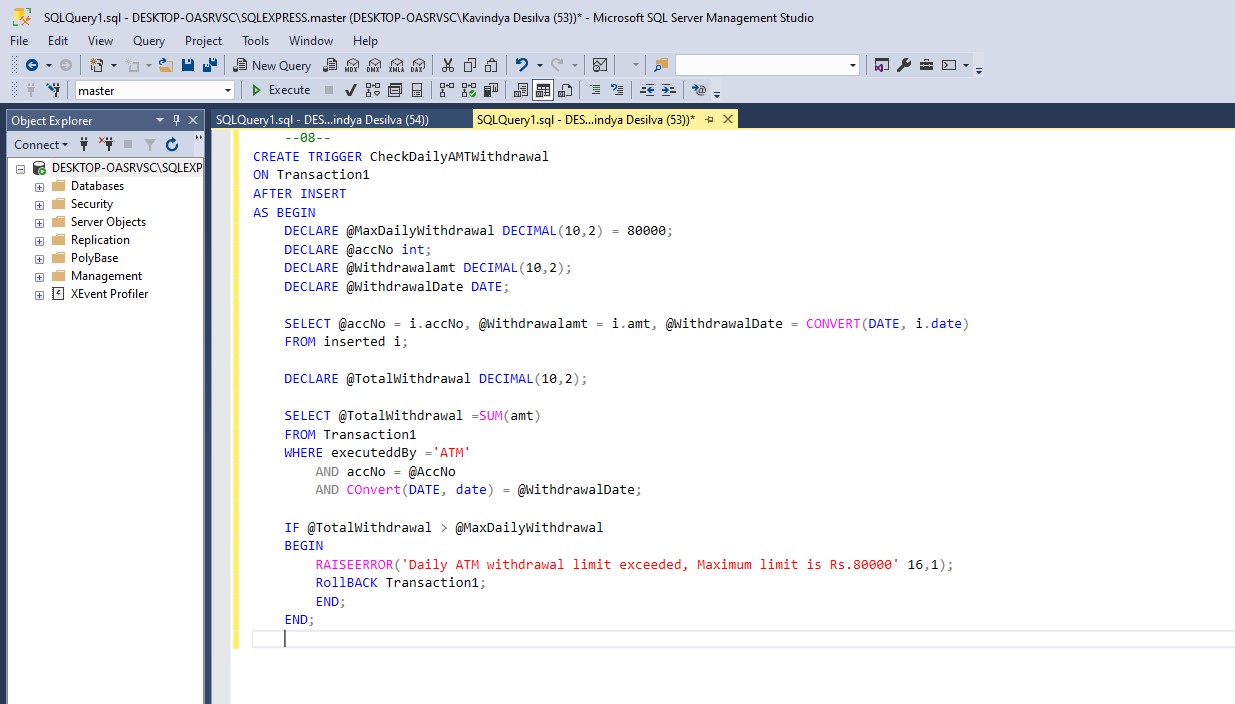
BEGIN

RAISEERROR('Daily ATM withdrawal limit exceeded, Maximum limit is Rs.80000' 16,1);

RollBACK Transaction1;

END;

END;



# 09)

CREATE TRIGGER UpdateAccBalance

ON Transaction1

AFTER INSERT

AS BEGIN

DECLARE @accNo int;

DECLARE @amt DECIMAL(10,2);

DECLARE @t\_type varchar(20);

SELECT @accNo = i.accNo, @amt = i.amt, @t\_type = i.executedBy

FROM inserted i;

IF @t\_type ='Credit'

Begin

EXEC UpdateAccBalance @accNo = @accNo, @amt= @amt, @operation ='Credit';

END

ELSE IF @t\_type = 'Debit'

BEGIN

EXEC UpdateAccBalance @accNo = @accNo, @amt = @amt, @operation ='Debit';

END

END;

