**Exercise 5: Triggers**

**Scenario 1:** Automatically update the last modified date when a customer's record is updated.

* + **Question:** Write a trigger **UpdateCustomerLastModified** that updates the LastModified column of the Customers table to the current date whenever a customer's record is updated.

**Ans:**

CREATE OR REPLACE TRIGGER UpdateCustomerLastModified

BEFORE UPDATE ON Customers

FOR EACH ROW

BEGIN

:NEW.LastModified := SYSDATE;

END UpdateCustomerLastModified;

/

**Scenario 2:** Maintain an audit log for all transactions.

* + **Question:** Write a trigger **LogTransaction** that inserts a record into an AuditLog table whenever a transaction is inserted into the Transactions table.

**Ans:**

**--First we need to creae AuditLog tavle**

BEGIN

EXECUTE IMMEDIATE '

CREATE TABLE AuditLog (

AuditID NUMBER PRIMARY KEY,

TransactionID NUMBER,

ActionType VARCHAR2(50),

ActionDate DATE,

OldBalance NUMBER,

NewBalance NUMBER

)';

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('An error occurred: ' || SQLERRM);

END;

/

**--Now we need to create sequence**

BEGIN

EXECUTE IMMEDIATE '

CREATE SEQUENCE AuditLog\_seq

START WITH 1

INCREMENT BY 1';

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('An error occurred: ' || SQLERRM);

END;

/

**--Now perform the trigger**

CREATE OR REPLACE TRIGGER LogTransaction

AFTER INSERT ON Transactions

FOR EACH ROW

DECLARE

oldBal NUMBER;

newBal NUMBER;

BEGIN

SELECT Balance INTO oldBal

FROM Accounts

WHERE AccountID = :NEW.AccountID;

IF :NEW.TransactionType = 'Deposit' THEN

newBal := oldBal + :NEW.Amount;

ELSE

newBal := oldBal - :NEW.Amount;

END IF;

INSERT INTO AuditLog (AuditID, TransactionID, ActionType, ActionDate, OldBalance, NewBalance)

VALUES (

AuditLog\_seq.NEXTVAL, -- Assume AuditLog\_seq is a sequence for generating AuditID

:NEW.TransactionID,

:NEW.TransactionType,

SYSDATE,

oldBal,

newBal

);

END LogTransaction;

/

**Scenario 3:** Enforce business rules on deposits and withdrawals.

* + **Question:** Write a trigger **CheckTransactionRules** that ensures withdrawals do not exceed the balance and deposits are positive before inserting a record into the Transactions table.

**Ans:**

CREATE OR REPLACE TRIGGER CheckTransactionRules

BEFORE INSERT ON Transactions

FOR EACH ROW

DECLARE

bal NUMBER;

BEGIN

SELECT Balance INTO bal

FROM Accounts

WHERE AccountID = :NEW.AccountID;

IF :NEW.TransactionType = 'Withdrawal' AND :NEW.Amount > bal THEN

RAISE\_APPLICATION\_ERROR(-20002, 'Insufficient funds for withdrawal.');

END IF;

IF :NEW.TransactionType = 'Deposit' AND :NEW.Amount <= 0 THEN

RAISE\_APPLICATION\_ERROR(-20003, 'Deposit amount must be positive.');

END IF;

END CheckTransactionRules;

/