**Exercise 5 – Triggers**

**Scenario 1: Automatically Update the Last Modified Date:**

CREATE OR REPLACE TRIGGER UpdateCustomerLastModified

BEFORE UPDATE ON Customers

FOR EACH ROW

BEGIN

-- Update the LastModified column to the current date

:NEW.LastModified := SYSDATE;

END UpdateCustomerLastModified;

/

**Scenario 2: Maintain an Audit Log for All Transactions**

CREATE OR REPLACE TRIGGER LogTransaction

AFTER INSERT ON Transactions

FOR EACH ROW

BEGIN

-- Insert a record into the AuditLog table

INSERT INTO AuditLog (TransactionID, AccountID, TransactionType, Amount, TransactionDate, LogDate)

VALUES (:NEW.TransactionID, :NEW.AccountID, :NEW.TransactionType, :NEW.Amount, :NEW.TransactionDate, SYSDATE);

END LogTransaction;

/

**Scenario 3: Enforce Business Rules on Deposits and Withdrawals**

CREATE OR REPLACE TRIGGER CheckTransactionRules

BEFORE INSERT ON Transactions

FOR EACH ROW

DECLARE

v\_Balance NUMBER;

BEGIN

-- Get the current balance of the account

SELECT Balance INTO v\_Balance FROM Accounts WHERE AccountID = :NEW.AccountID FOR UPDATE;

-- Enforce business rules

IF :NEW.TransactionType = 'WITHDRAWAL' THEN

IF :NEW.Amount > v\_Balance THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Error: Withdrawal amount exceeds the available balance.');

END IF;

ELSIF :NEW.TransactionType = 'DEPOSIT' THEN

IF :NEW.Amount <= 0 THEN

RAISE\_APPLICATION\_ERROR(-20002, 'Error: Deposit amount must be positive.');

END IF;

END IF;

END CheckTransactionRules;

/