**Exercise 5: Triggers**

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

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.

CREATE TABLE AuditLog (

AuditID NUMBER PRIMARY KEY,

TransactionID NUMBER,

ActionTime DATE,

ActionType VARCHAR2(20),

OldValue VARCHAR2(4000),

NewValue VARCHAR2(4000)

);

CREATE OR REPLACE TRIGGER LogTransaction

AFTER INSERT ON Transactions

FOR EACH ROW

BEGIN

INSERT INTO AuditLog (TransactionID, ActionTime, ActionType, OldValue, NewValue)

VALUES (:NEW.TransactionID, SYSDATE, 'INSERT', NULL, 'Amount: ' || :NEW.Amount || ', Type: ' || :NEW.TransactionType);

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\_current\_balance NUMBER;

BEGIN

-- For deposits, check that the amount is positive

IF :NEW.TransactionType = 'Deposit' THEN

IF :NEW.Amount <= 0 THEN

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

END IF;

END IF;

-- For withdrawals, check that the amount does not exceed the balance

IF :NEW.TransactionType = 'Withdrawal' THEN

SELECT Balance INTO v\_current\_balance

FROM Accounts

WHERE AccountID = :NEW.AccountID;

IF v\_current\_balance < :NEW.Amount THEN

RAISE\_APPLICATION\_ERROR(-20004, 'Insufficient funds for withdrawal.');

END IF;

END IF;

END CheckTransactionRules;

/