**EXERCISE 5 – TRIGGERS**

**SCENARIO 1 - AUTOMATICALLY UPDATE THE LAST MODIFIED DATE WHEN A CUSTOMER'S RECORD IS UPDATED.**

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

CREATE OR REPLACE TRIGGER UpdateCustomerLastModified

BEFORE UPDATE ON Customers

FOR EACH ROW

BEGIN

:NEW.LastModified := SYSDATE;

END;

/

**SCENARIO 2 - MAINTAIN AN AUDIT LOG FOR ALL TRANSACTIONS**

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

CREATE OR REPLACE TRIGGER LogTransaction

AFTER INSERT ON Transactions

FOR EACH ROW

BEGIN

INSERT INTO AuditLog VALUES ( AuditLog\_SEQ.NEXTVAL, :NEW.TransactionID, SYSDATE,

'INSERT', 'Transaction inserted with AccountID ' || :NEW.AccountID ||

', Amount ' || :NEW.Amount || ', Type ' || :NEW.TransactionType

);

END;

/

-- testing the logging

INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType)

VALUES ( 4, 3, SYSDATE, 1000, 'Deposit' );

INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType)

VALUES ( 5, 4, SYSDATE, 1000, 'Withdraw' );

SELECT \* FROM AuditLog;

**SCENARIO 3 - ENFORCE BUSINESS RULES ON DEPOSITS AND WITHDRAWALS.**

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

CREATE OR REPLACE TRIGGER CheckTransactionRules

BEFORE INSERT ON Transactions

FOR EACH ROW

DECLARE

v\_balance NUMBER;

BEGIN

SELECT Balance INTO v\_balance FROM Accounts WHERE AccountID = :NEW.AccountID;

IF :NEW.TransactionType = 'Withdrawal' THEN

IF :NEW.Amount > v\_balance THEN

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

END IF;

END IF;

IF :NEW.TransactionType = 'Deposit' THEN

IF :NEW.Amount <= 0 THEN

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

END IF;

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20003, 'Account ID does not exist.');

WHEN OTHERS THEN

RAISE\_APPLICATION\_ERROR(-20004, 'Error: ' || SQLERRM);

END;

/

-- checking the trigger

-- Test with a withdrawal that exceeds the balance

INSERT INTO Transactions VALUES (6, 3, SYSDATE, 3000, 'Withdrawal' );

-- This should raise an error if the balance is less than 3000

-- Test with a withdrawal that does not exceeds the balance

INSERT INTO Transactions VALUES (6, 3, SYSDATE, 300, 'Withdrawal' );

-- Test with a deposit that is zero

INSERT INTO Transactions VALUES (7, 5, SYSDATE, 0, 'Deposit' );

-- This should raise an error because the deposit amount is zero