**Exercise 5: Triggers**

**Step 1: Update Last Modified Date**

CREATE OR REPLACE TRIGGER UpdateCustomerLastModified

BEFORE UPDATE ON customers

FOR EACH ROW

BEGIN

:NEW.LastModified := SYSDATE;

END UpdateCustomerLastModified;

/

**Step 2: Maintain an Audit Log**

**1.Create the AuditLog table:**

CREATE TABLE AuditLog (

audit\_id NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,

transaction\_id NUMBER,

account\_id NUMBER,

transaction\_type VARCHAR2(10),

amount NUMBER,

transaction\_date DATE,

log\_date DATE

);

**2. Create the LogTransaction trigger:**

CREATE OR REPLACE TRIGGER LogTransaction

AFTER INSERT ON Transactions

FOR EACH ROW

BEGIN

INSERT INTO AuditLog (transaction\_id, account\_id, transaction\_type, amount, transaction\_date, log\_date)

VALUES (:NEW.transaction\_id, :NEW.account\_id, :NEW.transaction\_type, :NEW.amount, :NEW.transaction\_date, SYSDATE);

END LogTransaction;

/

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

1. **Create the Transactions table:**

CREATE TABLE Transactions (

transaction\_id NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,

account\_id NUMBER,

transaction\_type VARCHAR2(10),

amount NUMBER,

transaction\_date DATE,

FOREIGN KEY (account\_id) REFERENCES accounts(account\_id)

);

1. **Create the CheckTransactionRules trigger:**

CREATE OR REPLACE TRIGGER CheckTransactionRules

BEFORE INSERT ON Transactions

FOR EACH ROW

DECLARE

v\_balance NUMBER;

BEGIN

-- Check if the transaction is a withdrawal

IF :NEW.transaction\_type = 'Withdrawal' THEN

-- Retrieve the current balance of the account

SELECT balance INTO v\_balance

FROM accounts

WHERE account\_id = :NEW.account\_id

FOR UPDATE;

-- Ensure the withdrawal amount does not exceed the balance

IF :NEW.amount > v\_balance THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Insufficient balance for withdrawal');

END IF;

-- Update the account balance

UPDATE accounts

SET balance = balance - :NEW.amount

WHERE account\_id = :NEW.account\_id;

ELSIF :NEW.transaction\_type = 'Deposit' THEN

-- Ensure the deposit amount is positive

IF :NEW.amount <= 0 THEN

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

END IF;

-- Update the account balance

UPDATE accounts

SET balance = balance + :NEW.amount

WHERE account\_id = :NEW.account\_id;

END IF;

END CheckTransactionRules;

/

**Example Data Insertion Scripts**

-- Creating tables

CREATE TABLE customers (

customer\_id NUMBER PRIMARY KEY,

customer\_name VARCHAR2(100),

age NUMBER,

balance NUMBER,

loan\_interest\_rate NUMBER,

LastModified DATE

);

CREATE TABLE accounts (

account\_id NUMBER PRIMARY KEY,

balance NUMBER

);

-- Insert example data

INSERT INTO customers (customer\_id, customer\_name, age, balance, loan\_interest\_rate) VALUES (1, 'John Doe', 65, 5000, 5);

INSERT INTO customers (customer\_id, customer\_name, age, balance, loan\_interest\_rate) VALUES (2, 'Jane Smith', 55, 15000, 6);

INSERT INTO accounts (account\_id, balance) VALUES (1, 1000);

INSERT INTO accounts (account\_id, balance) VALUES (2, 2000);

COMMIT;