**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.

**Code:**

CREATE OR REPLACE TRIGGER UpdateCustomerLastModified

BEFORE UPDATE ON Customers

FOR EACH ROW

BEGIN

:NEW.LastModified := SYSDATE;

END;

/

SELECT CustomerID, Name, LastModified FROM Customers WHERE CustomerID = 1;

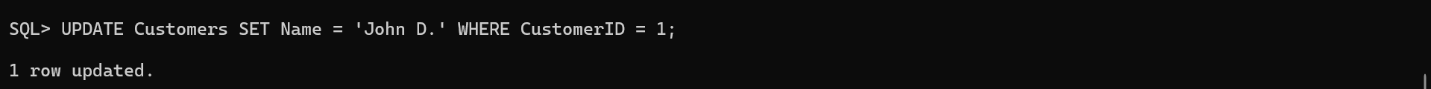
UPDATE Customers SET Name = 'John D.' WHERE CustomerID = 1;

SELECT CustomerID, Name, LastModified FROM Customers WHERE CustomerID = 1;

**Output:**

**A black screen with a black background

AI-generated content may be incorrect.A black screen with white text

AI-generated content may be incorrect.A black screen with white text

AI-generated content may be incorrect.**

**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.

**Code:**

CREATE TABLE AuditLog (

LogID NUMBER GENERATED ALWAYS AS IDENTITY PRIMARY KEY,

TransactionID NUMBER,

AccountID NUMBER,

Action VARCHAR2(20),

ActionDate DATE

);

CREATE OR REPLACE TRIGGER LogTransaction

AFTER INSERT ON Transactions

FOR EACH ROW

BEGIN

INSERT INTO AuditLog (TransactionID, AccountID, Action, ActionDate)

VALUES (:NEW.TransactionID, :NEW.AccountID, 'INSERT', SYSDATE);

END;

/

INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType) VALUES (3, 1, SYSDATE, 100, 'Deposit');

SELECT \* FROM AuditLog ORDER BY LogID DESC;

**Output:**

**A black screen with white text

AI-generated content may be incorrect.A black screen with white text

AI-generated content may be incorrect.A black screen with white text

AI-generated content may be incorrect.**

**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.

**Code:**

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 account balance.');

END IF;

ELSIF :NEW.TransactionType = 'Deposit' THEN

IF :NEW.Amount <= 0 THEN

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

END IF;

END IF;

END;

/

INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType) VALUES (4, 1, SYSDATE, 200, 'Deposit');

INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType) VALUES (5, 1, SYSDATE, -100, 'Deposit');

INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType) VALUES (6, 1, SYSDATE, 50, 'Withdrawal');

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

VALUES (7, 1, SYSDATE, 999999, 'Withdrawal');

**Output:**

**A screenshot of a computer

AI-generated content may be incorrect.A black screen with white text

AI-generated content may be incorrect.A screen shot of a computer

AI-generated content may be incorrect.**