**Scenario 1: Update LastModified on Customer Update**

CREATE OR REPLACE TRIGGER UpdateCustomerLastModified

BEFORE UPDATE ON Customers

FOR EACH ROW

BEGIN

:NEW.LastModified := SYSDATE;

END;

**Example Execution:**

UPDATE Customers SET Name = 'John Updated' WHERE CustomerID = 1;

**Output:**

text

Copy code

-- Name updated

-- LastModified automatically set to current date

**Scenario 2: Log Transactions to Audit Table**

**Assumed Table:**

CREATE TABLE AuditLog (

AuditID NUMBER GENERATED ALWAYS AS IDENTITY,

TransactionID NUMBER,

AuditDate DATE,

Action VARCHAR2(100)

);

CREATE OR REPLACE TRIGGER LogTransaction

AFTER INSERT ON Transactions

FOR EACH ROW

BEGIN

INSERT INTO AuditLog (TransactionID, AuditDate, Action)

VALUES (:NEW.TransactionID, SYSDATE, 'Transaction Inserted');

END;

**Example Insert:**

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

VALUES (3, 1, SYSDATE, 400, 'Deposit');

**Output:**

-- New transaction added

-- Entry in AuditLog: (TransactionID = 3, Action = 'Transaction Inserted')

**Scenario 3: Enforce Transaction Rules**

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' AND :NEW.Amount > v\_balance THEN

RAISE\_APPLICATION\_ERROR(-20002, 'Withdrawal exceeds balance');

ELSIF :NEW.TransactionType = 'Deposit' AND :NEW.Amount <= 0 THEN

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

END IF;

END;

**Example Insert (Invalid):**

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

VALUES (4, 1, SYSDATE, 10000, 'Withdrawal');

**Output:**

ORA-20002: Withdrawal exceeds balance