PL/SQL Programming

SuperSet ID:6412063

Exercise 5: Triggers

Scenario 1:

CREATE OR REPLACE TRIGGER UpdateCustomerLastModified

BEFORE UPDATE ON Customers

FOR EACH ROW

BEGIN

:NEW.LastModified := SYSDATE;

END;

/

UPDATE Customers

SET Balance = Balance + 1000

WHERE CustomerID = 1;

SELECT CustomerID, Balance, LastModified

FROM Customers

WHERE CustomerID = 1;

Output:

A black screen with a black background

Description automatically generated

Scenario 2:

CREATE TABLE AuditLog (

AuditID NUMBER GENERATED BY DEFAULT ON NULL AS IDENTITY PRIMARY KEY,

TransactionID NUMBER,

Action VARCHAR2(20),

ActionDate DATE

);

CREATE OR REPLACE TRIGGER LogTransaction

AFTER INSERT ON Transactions

FOR EACH ROW

BEGIN

INSERT INTO AuditLog (TransactionID, Action, ActionDate)

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

END;

/

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

VALUES (1001, 101, SYSDATE, 500, 'Deposit');

SELECT \* FROM AuditLog WHERE TransactionID = 1001;

Output:

A black background with white text

Description automatically generated

Scenario 3:

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

RAISE\_APPLICATION\_ERROR(-20001, 'Deposit amount must be positive.');

END IF;

IF :NEW.TransactionType = 'Withdrawal' AND :NEW.Amount > v\_balance THEN

RAISE\_APPLICATION\_ERROR(-20002, 'Insufficient balance for withdrawal.');

END IF;

END;

/

Output:

A screen shot of a computer

Description automatically generated