**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:**

BEGIN

EXECUTE IMMEDIATE 'DROP TABLE customers';

EXCEPTION

WHEN OTHERS THEN NULL;

END;

/

CREATE TABLE customers (

customer\_id NUMBER PRIMARY KEY,

name VARCHAR2(100),

dob DATE,

lastmodified DATE

);

INSERT INTO customers VALUES (1, 'Meena', DATE '2000-01-01', NULL);

INSERT INTO customers VALUES (2, 'Sri', DATE '1999-12-12', NULL);

COMMIT;

CREATE OR REPLACE TRIGGER UpdateCustomerLastModified

BEFORE UPDATE ON customers

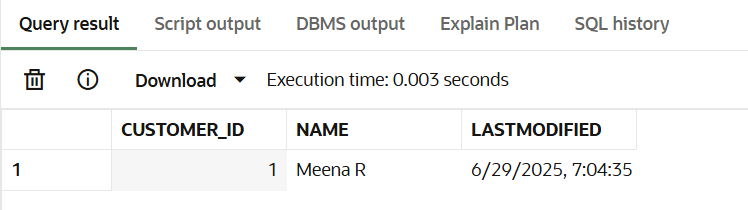
FOR EACH ROW

BEGIN

:NEW.lastmodified := SYSDATE;

END;

/

**OUTPUT:** ****

**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:**

BEGIN

EXECUTE IMMEDIATE 'DROP TABLE transactions';

EXCEPTION

WHEN OTHERS THEN NULL;

END;

/

BEGIN

EXECUTE IMMEDIATE 'DROP TABLE auditlog';

EXCEPTION

WHEN OTHERS THEN NULL;

END;

/

CREATE TABLE transactions (

txn\_id NUMBER PRIMARY KEY,

account\_id NUMBER,

txn\_type VARCHAR2(20),

amount NUMBER(10,2),

txn\_date DATE

);

CREATE TABLE auditlog (

log\_id NUMBER GENERATED ALWAYS AS IDENTITY PRIMARY KEY,

txn\_id NUMBER,

action VARCHAR2(50),

log\_time DATE

);

CREATE OR REPLACE TRIGGER LogTransaction

AFTER INSERT ON transactions

FOR EACH ROW

BEGIN

INSERT INTO auditlog (txn\_id, action, log\_time)

VALUES (:NEW.txn\_id, 'Transaction Inserted', SYSDATE);

END;

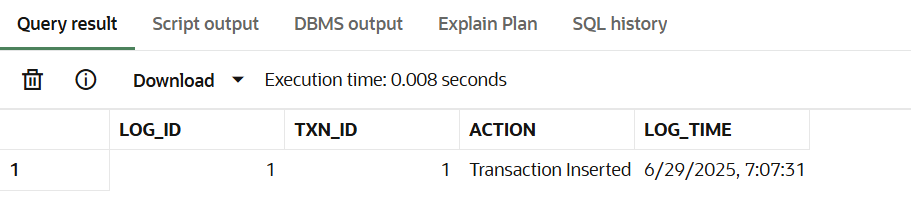
/

INSERT INTO transactions (txn\_id, account\_id, txn\_type, amount, txn\_date)

VALUES (1, 101, 'deposit', 1000, SYSDATE);

SELECT \* FROM auditlog;

**OUTPUT:**

****

**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:**

BEGIN

EXECUTE IMMEDIATE 'DROP TABLE accounts';

EXCEPTION

WHEN OTHERS THEN NULL;

END;

/

CREATE TABLE accounts (

accountid NUMBER PRIMARY KEY,

balance NUMBER(10,2)

);

INSERT INTO accounts VALUES (101, 800);

INSERT INTO accounts VALUES (102, 700);

COMMIT;

CREATE OR REPLACE TRIGGER CheckTransactionRules

BEFORE INSERT ON transactions

FOR EACH ROW

DECLARE

acc\_balance NUMBER;

BEGIN

SELECT balance INTO acc\_balance

FROM accounts

WHERE accountid = :NEW.account\_id;

IF :NEW.txn\_type = 'withdraw' AND :NEW.amount > acc\_balance THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Withdrawal exceeds account balance.');

END IF;

IF :NEW.txn\_type = 'deposit' AND :NEW.amount <= 0 THEN

RAISE\_APPLICATION\_ERROR(-20002, 'Deposit amount must be greater than 0.');

END IF;

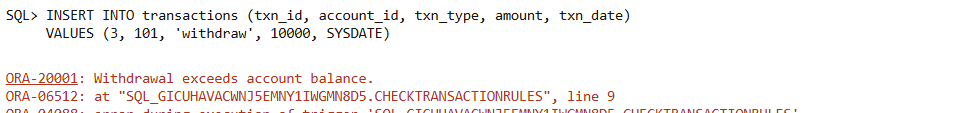
END;

/

**OUTPUT:**

**I tried to withdraw ₹10,000 from account 101, but the balance is only ₹800.**

**The trigger CheckTransactionRules correctly blocked the transaction and raised this error:**

****