**Scenario 1: Update Last Modified Date for Customer Records**

```sql

CREATE OR REPLACE TRIGGER UpdateCustomerLastModified

BEFORE UPDATE ON customers

FOR EACH ROW

BEGIN

:NEW.last\_modified := SYSDATE;

END;

/

```

**Scenario 2: Maintain an Audit Log for Transactions**

```sql

CREATE OR REPLACE TRIGGER LogTransaction

AFTER INSERT ON transactions

FOR EACH ROW

BEGIN

INSERT INTO audit\_log (transaction\_id, transaction\_date, action)

VALUES (:NEW.transaction\_id, :NEW.transaction\_date, 'INSERT');

END;

/

```

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

```sql

CREATE OR REPLACE TRIGGER CheckTransactionRules

BEFORE INSERT ON transactions

FOR EACH ROW

BEGIN

IF :NEW.transaction\_type = 'WITHDRAWAL' AND :NEW.amount > (SELECT balance FROM accounts WHERE account\_id = :NEW.account\_id) THEN

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

ELSIF :NEW.transaction\_type = 'DEPOSIT' AND :NEW.amount <= 0 THEN

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

END IF;

END;

/

```