**PL/SQL Stored Procedures for Banking Operations**

**Scenario 1: Monthly Interest Processing**

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest AS

v\_interest\_rate NUMBER := 0.01; -- 1% monthly interest

v\_updated\_count NUMBER := 0;

BEGIN

-- Apply interest to all savings accounts

UPDATE accounts

SET balance = balance \* (1 + v\_interest\_rate)

WHERE account\_type = 'SAVINGS';

v\_updated\_count := SQL%ROWCOUNT;

-- Record transaction in audit log

INSERT INTO interest\_transactions (transaction\_date, interest\_rate, accounts\_affected)

VALUES (SYSDATE, v\_interest\_rate, v\_updated\_count);

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Applied monthly interest to ' || v\_updated\_count || ' savings accounts');

EXCEPTION

WHEN OTHERS THEN

ROLLBACK;

DBMS\_OUTPUT.PUT\_LINE('Error processing interest: ' || SQLERRM);

-- Log error to error table

INSERT INTO error\_log (error\_date, procedure\_name, error\_message)

VALUES (SYSDATE, 'PROCESSMONTHLYINTEREST', SQLERRM);

END ProcessMonthlyInterest;

/

**Execution:**

sql

Copy

Download

EXEC ProcessMonthlyInterest;

**Scenario 2: Employee Bonus Calculation**

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(

p\_department\_id IN NUMBER,

p\_bonus\_percent IN NUMBER,

p\_updated\_count OUT NUMBER

) AS

BEGIN

-- Validate bonus percentage

IF p\_bonus\_percent < 0 OR p\_bonus\_percent > 50 THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Bonus percentage must be between 0 and 50');

END IF;

-- Update salaries with bonus

UPDATE employees

SET salary = salary \* (1 + p\_bonus\_percent/100),

last\_bonus\_date = SYSDATE

WHERE department\_id = p\_department\_id;

p\_updated\_count := SQL%ROWCOUNT;

COMMIT;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('No employees found in department ' || p\_department\_id);

p\_updated\_count := 0;

WHEN OTHERS THEN

ROLLBACK;

DBMS\_OUTPUT.PUT\_LINE('Error updating bonuses: ' || SQLERRM);

p\_updated\_count := -1; -- Error indicator

END UpdateEmployeeBonus;

/

**Execution:**

DECLARE

v\_count NUMBER;

BEGIN

UpdateEmployeeBonus(p\_department\_id => 10, p\_bonus\_percent => 15, p\_updated\_count => v\_count);

DBMS\_OUTPUT.PUT\_LINE('Updated ' || v\_count || ' employees');

END;

/

**Scenario 3: Secure Funds Transfer**

CREATE OR REPLACE PROCEDURE TransferFunds(

p\_from\_account IN NUMBER,

p\_to\_account IN NUMBER,

p\_amount IN NUMBER,

p\_transaction\_id OUT NUMBER,

p\_status OUT VARCHAR2

) AS

v\_from\_balance NUMBER;

v\_to\_account\_exists NUMBER;

BEGIN

-- Check if amount is positive

IF p\_amount <= 0 THEN

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

END IF;

-- Verify source account balance (with FOR UPDATE to lock row)

SELECT balance INTO v\_from\_balance

FROM accounts

WHERE account\_id = p\_from\_account

FOR UPDATE;

IF v\_from\_balance < p\_amount THEN

p\_status := 'FAILED: Insufficient funds';

RETURN;

END IF;

-- Verify destination account exists

SELECT COUNT(\*) INTO v\_to\_account\_exists

FROM accounts

WHERE account\_id = p\_to\_account;

IF v\_to\_account\_exists = 0 THEN

p\_status := 'FAILED: Destination account not found';

RETURN;

END IF;

-- Perform transfer

UPDATE accounts SET balance = balance - p\_amount

WHERE account\_id = p\_from\_account;

UPDATE accounts SET balance = balance + p\_amount

WHERE account\_id = p\_to\_account;

-- Record transaction

INSERT INTO transactions (

transaction\_id, from\_account, to\_account, amount, transaction\_date

) VALUES (

transaction\_seq.NEXTVAL, p\_from\_account, p\_to\_account, p\_amount, SYSDATE

) RETURNING transaction\_id INTO p\_transaction\_id;

COMMIT;

p\_status := 'SUCCESS';

EXCEPTION

WHEN OTHERS THEN

ROLLBACK;

p\_status := 'FAILED: ' || SUBSTR(SQLERRM, 1, 200);

p\_transaction\_id := -1;

END TransferFunds;

/

**Execution:**

DECLARE

v\_transaction\_id NUMBER;

v\_status VARCHAR2(200);

BEGIN

TransferFunds(

p\_from\_account => 1001,

p\_to\_account => 2002,

p\_amount => 500,

p\_transaction\_id => v\_transaction\_id,

p\_status => v\_status

);

DBMS\_OUTPUT.PUT\_LINE('Transaction ' || v\_transaction\_id || ': ' || v\_status);

END;

/