**Exercise 1: Control Structures**

**Scenario 1:** The bank wants to apply a discount to loan interest rates for customers above 60 years old.

**Solution:**

BEGIN

FOR rec IN (SELECT customer\_id FROM customers WHERE age > 60) LOOP

UPDATE loans

SET interest\_rate = interest\_rate - 1

WHERE customer\_id = rec.customer\_id;

END LOOP;

COMMIT;

END;

**Scenario 2:** A customer can be promoted to VIP status based on their balance.

**Solution:**

BEGIN

FOR rec IN (SELECT customer\_id FROM customers WHERE balance > 10000) LOOP

UPDATE customers

SET IsVIP = 'TRUE'

WHERE customer\_id = rec.customer\_id;

END LOOP;

COMMIT;

END;

**Scenario 3:** The bank wants to send reminders to customers whose loans are due within the next 30 days.

**Solution:**

DECLARE

CURSOR loan\_cursor IS

SELECT l.customer\_id, c.f\_name, l.due\_date

FROM loans l

JOIN customers c ON l.customer\_id = c.customer\_id

WHERE l.due\_date <= SYSDATE + 30;

loan\_rec loan\_cursor%ROWTYPE;

BEGIN

OPEN loan\_cursor;

LOOP

FETCH loan\_cursor INTO loan\_rec;

EXIT WHEN loan\_cursor%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('Reminder: ' || loan\_rec.f\_name ||

', your loan is due on ' || loan\_rec.due\_date);

END LOOP;

CLOSE loan\_cursor;

END;

**Scenario 1:** The bank needs to process monthly interest for all savings accounts.

**Solution:**

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

UPDATE accounts

SET balance = balance + (balance \* 0.01)

WHERE account\_type = 'savings';

COMMIT;

END;

**Scenario 2:** The bank wants to implement a bonus scheme for employees based on their performance.

**Solution:**

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(

dept\_id IN NUMBER,

bonus\_percent IN NUMBER

) IS

BEGIN

UPDATE employees

SET salary = salary + (salary \* bonus\_percent / 100)

WHERE department\_id = dept\_id;

COMMIT;

END;

**Scenario 3:** Customers should be able to transfer funds between their accounts.

**Solution:**

CREATE OR REPLACE PROCEDURE TransferFunds(

from\_account IN NUMBER,

to\_account IN NUMBER,

amount IN NUMBER ) IS

from\_balance NUMBER;

BEGIN

SELECT balance INTO from\_balance FROM accounts WHERE account\_id = from\_account;

IF from\_balance >= amount THEN

UPDATE accounts

SET balance = balance - amount

WHERE account\_id = from\_account;

UPDATE accounts

SET balance = balance + amount

WHERE account\_id = to\_account;

COMMIT;

ELSE

DBMS\_OUTPUT.PUT\_LINE('Insufficient balance in source account.');

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Account not found.');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE(SQLERRM);

END;