**Exercise 1: Control Structures**

**Scenario 1: Reduce Interest Rate by 1% for Senior Customers**

BEGIN

FOR rec IN (

SELECT customer\_id, age

FROM Customers

) LOOP

IF rec.age > 60 THEN

UPDATE Loans

SET interest\_rate = interest\_rate - 1

WHERE customer\_id = rec.customer\_id;

END IF;

END LOOP;

COMMIT;

END;

/

**Scenario 2: Upgrade Customer to VIP for High Balances**

BEGIN

FOR cust IN (

SELECT customer\_id, balance

FROM Customers

) LOOP

IF cust.balance > 10000 THEN

UPDATE Customers

SET IsVIP = 'TRUE'

WHERE customer\_id = cust.customer\_id;

END IF;

END LOOP;

COMMIT;

END;

/

**Scenario 3: Alert for Loans Due in the Next 30 Days**

DECLARE

upcoming\_due DATE := SYSDATE + 30;

BEGIN

FOR reminder IN (

SELECT l.loan\_id, l.customer\_id, c.name, l.due\_date

FROM Loans l

INNER JOIN Customers c ON l.customer\_id = c.customer\_id

WHERE l.due\_date BETWEEN SYSDATE AND upcoming\_due

) LOOP

DBMS\_OUTPUT.PUT\_LINE('Alert: Loan ' || reminder.loan\_id ||

' (Customer: ' || reminder.name ||

') is due on ' || TO\_CHAR(reminder.due\_date, 'DD-MON-YYYY'));

END LOOP;

END;

/

**Exercise 3: Stored Procedures**

**Scenario 1: Apply Monthly Interest to Savings Accounts**

CREATE OR REPLACE PROCEDURE ApplyMonthlyInterest AS

BEGIN

UPDATE Accounts

SET balance = balance + (balance \* 0.01)

WHERE account\_type = 'SAVINGS';

COMMIT;

END;

/

**Scenario 2: Add Bonus to Employee Salaries**

CREATE OR REPLACE PROCEDURE AdjustBonus (

p\_dept\_id IN NUMBER,

p\_bonus IN NUMBER

) AS

BEGIN

UPDATE Employees

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

WHERE department\_id = p\_dept\_id;

COMMIT;

END;

/

**Scenario 3: Fund Transfer Between Two Accounts with Error Handling**

CREATE OR REPLACE PROCEDURE ExecuteTransfer (

sender\_acc IN NUMBER,

receiver\_acc IN NUMBER,

amount\_to\_transfer IN NUMBER

) AS

ex\_low\_balance EXCEPTION;

sender\_balance NUMBER;

BEGIN

-- Lock and retrieve sender's balance

SELECT balance INTO sender\_balance

FROM Accounts

WHERE account\_id = sender\_acc

FOR UPDATE;

IF sender\_balance < amount\_to\_transfer THEN

RAISE ex\_low\_balance;

END IF;

-- Deduct from sender

UPDATE Accounts

SET balance = balance - amount\_to\_transfer

WHERE account\_id = sender\_acc;

-- Credit to receiver

UPDATE Accounts

SET balance = balance + amount\_to\_transfer

WHERE account\_id = receiver\_acc;

COMMIT;

EXCEPTION

WHEN ex\_low\_balance THEN

ROLLBACK;

DBMS\_OUTPUT.PUT\_LINE('Error: Insufficient funds for transfer.');

WHEN OTHERS THEN

ROLLBACK;

DBMS\_OUTPUT.PUT\_LINE('Unexpected error occurred: ' || SQLERRM);

END;

/