**Exercise 1: Control Structures**

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

**PL/SQL Query:**

BEGIN

FOR r IN (

SELECT l.LoanID

FROM loans l

JOIN customers c ON l.CustomerID = c.CustomerID

WHERE c.Age > 60

)

LOOP

UPDATE loans

SET InterestRate = InterestRate - 1

WHERE LoanID = r.LoanID;

END LOOP;

END;

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

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

**PL/SQL Query:**

BEGIN

UPDATE customers

SET IsVIP = 'Y'

WHERE Balance > 10000;

END;

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

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

**PL/SQL Query:**

BEGIN

FOR r IN (

SELECT c.Name, l.DueDate

FROM loans l

JOIN customers c ON l.CustomerID = c.CustomerID

WHERE l.DueDate BETWEEN SYSDATE AND SYSDATE + 30

)

LOOP

DBMS\_OUTPUT.PUT\_LINE('Reminder: Dear ' || r.Name || ', your loan is due on ' || TO\_CHAR(r.DueDate, 'DD-MON-YYYY'));

END LOOP;

END;

A screenshot of a computer

AI-generated content may be incorrect.

**Exercise 3: Stored Procedures**

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

**PL/SQL Query:**

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

    UPDATE accounts

    SET Balance = Balance + (Balance \* 0.01)

    WHERE AccountType = 'savings';

END;

/

BEGIN

    ProcessMonthlyInterest;

END;

/

SELECT \* FROM accounts;

**A screenshot of a computer

AI-generated content may be incorrect.**

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

**PL/SQL Query:**

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(

    deptName IN VARCHAR2,

    bonusPercent IN NUMBER

) IS

BEGIN

    UPDATE employees

    SET Salary = Salary + (Salary \* bonusPercent / 100)

    WHERE Department = deptName;

END;

/

BEGIN

    UpdateEmployeeBonus('IT', 10);

END;

/SELECT \* FROM employees;

A screenshot of a computer

AI-generated content may be incorrect.

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

**PL/SQL Query:**

CREATE OR REPLACE PROCEDURE TransferFunds(

    fromAccount IN INT,

    toAccount IN INT,

    amount IN NUMBER

) IS

    insufficient\_balance EXCEPTION;

    source\_balance NUMBER;

BEGIN

    SELECT Balance INTO source\_balance FROM accounts WHERE AccountID = fromAccount FOR UPDATE;

    IF source\_balance < amount THEN

        RAISE insufficient\_balance;

    END IF;

    UPDATE accounts SET Balance = Balance - amount WHERE AccountID = fromAccount;

    UPDATE accounts SET Balance = Balance + amount WHERE AccountID = toAccount;

EXCEPTION

    WHEN insufficient\_balance THEN

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

    WHEN NO\_DATA\_FOUND THEN

        DBMS\_OUTPUT.PUT\_LINE('Error: Invalid account ID.');

END;

/

SET SERVEROUTPUT ON;

BEGIN

    TransferFunds(101, 102, 1000);

END;

/

SELECT \* FROM accounts;

A screenshot of a computer

AI-generated content may be incorrect.