**Exercise 4: Functions**

**Scenario 1:** Calculate the age of customers for eligibility checks.

**PL/SQL block:**

CREATE OR REPLACE FUNCTION CalculateAge (

p\_dob IN DATE

) RETURN NUMBER AS

v\_age NUMBER;

BEGIN

-- Calculate the age in years

v\_age := TRUNC(MONTHS\_BETWEEN(SYSDATE, p\_dob) / 12);

RETURN v\_age;

END;

/

**Scenario 2:** The bank needs to compute the monthly installment for a loan.

**PL/SQL block:**

CREATE OR REPLACE FUNCTION CalculateMonthlyInstallment (

p\_loan\_amount IN NUMBER,

p\_annual\_interest\_rate IN NUMBER,

p\_duration\_years IN NUMBER

) RETURN NUMBER AS

v\_monthly\_installment NUMBER;

v\_monthly\_interest\_rate NUMBER;

v\_total\_payments NUMBER;

BEGIN

-- Convert annual interest rate to monthly interest rate

v\_monthly\_interest\_rate := p\_annual\_interest\_rate / 12 / 100;

-- Total number of monthly payments

v\_total\_payments := p\_duration\_years \* 12;

-- Calculate the monthly installment using the formula for an amortizing loan

IF v\_monthly\_interest\_rate > 0 THEN

v\_monthly\_installment := p\_loan\_amount \* (v\_monthly\_interest\_rate \* POWER(1 + v\_monthly\_interest\_rate, v\_total\_payments)) /

(POWER(1 + v\_monthly\_interest\_rate, v\_total\_payments) - 1);

ELSE

v\_monthly\_installment := p\_loan\_amount / v\_total\_payments;

END IF;

RETURN v\_monthly\_installment;

END;

/

**Scenario 3:** Check if a customer has sufficient balance before making a transaction.

**PL/SQL block:**

CREATE OR REPLACE FUNCTION HasSufficientBalance (

p\_account\_id IN NUMBER,

p\_amount IN NUMBER

) RETURN BOOLEAN AS

v\_balance NUMBER;

BEGIN

-- Retrieve the balance of the specified account

SELECT Balance INTO v\_balance FROM Accounts WHERE AccountID = p\_account\_id;

-- Check if the balance is sufficient

IF v\_balance >= p\_amount THEN

RETURN TRUE;

ELSE

RETURN FALSE;

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN FALSE;

WHEN OTHERS THEN

-- Handle other unexpected errors

RETURN FALSE;

END;

/