**Exercise 4: Functions**

**Scenario 1: Calculate the Age of Customers for Eligibility Checks**

CREATE OR REPLACE FUNCTION CalculateAge (

p\_date\_of\_birth IN DATE

) RETURN NUMBER IS

v\_age NUMBER;

BEGIN

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

RETURN v\_age;

END CalculateAge;

/

**Scenario 2: Compute the Monthly Installment for a Loan**

CREATE OR REPLACE FUNCTION CalculateMonthlyInstallment (

p\_loan\_amount IN NUMBER,

p\_interest\_rate IN NUMBER,

p\_loan\_duration\_years IN NUMBER

) RETURN NUMBER IS

v\_monthly\_interest\_rate NUMBER;

v\_number\_of\_payments NUMBER;

v\_monthly\_installment NUMBER;

BEGIN

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

v\_number\_of\_payments := p\_loan\_duration\_years \* 12;

IF v\_monthly\_interest\_rate = 0 THEN

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

ELSE

v\_monthly\_installment := p\_loan\_amount \* v\_monthly\_interest\_rate /

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

END IF;

RETURN v\_monthly\_installment;

END CalculateMonthlyInstallment;

/

**Scenario 3: Check if a Customer Has Sufficient Balance Before Making a Transaction**

CREATE OR REPLACE FUNCTION HasSufficientBalance (

p\_account\_id IN accounts.account\_id%TYPE,

p\_amount IN NUMBER

) RETURN BOOLEAN IS

v\_balance accounts.balance%TYPE;

BEGIN

SELECT balance INTO v\_balance

FROM accounts

WHERE account\_id = p\_account\_id;

IF v\_balance >= p\_amount THEN

RETURN TRUE;

ELSE

RETURN FALSE;

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN FALSE;

END HasSufficientBalance;/