**Exercise 4: Functions**

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

**Question:** Write a function CalculateAge that takes a customer's date of birth as input and returns their age in years.

**CODE**

CREATE OR REPLACE FUNCTION CalculateAge (

    date\_of\_birth IN DATE

) RETURN NUMBER

IS

    age NUMBER;

BEGIN

    age := FLOOR(MONTHS\_BETWEEN(SYSDATE, date\_of\_birth) / 12);

    RETURN age;

END CalculateAge;

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

**Question:** Write a function **CalculateMonthlyInstallment** that takes the loan amount, interest rate, and loan duration in years as input and returns the monthly installment amount.

**CODE**

CREATE OR REPLACE FUNCTION CalculateMonthlyInstallment (

    loan\_amount IN NUMBER,

    annual\_interest\_rate IN NUMBER,

    loan\_duration\_years IN NUMBER

) RETURN NUMBER

IS

    monthly\_interest\_rate NUMBER;

    number\_of\_payments NUMBER;

    monthly\_installment NUMBER;

BEGIN

    monthly\_interest\_rate := annual\_interest\_rate / 12 / 100;

    number\_of\_payments := loan\_duration\_years \* 12;

    IF monthly\_interest\_rate = 0 THEN

        monthly\_installment := loan\_amount / number\_of\_payments;

    ELSE

        monthly\_installment := loan\_amount \* monthly\_interest\_rate /

            (1 - POWER(1 + monthly\_interest\_rate, -number\_of\_payments));

    END IF;

    RETURN monthly\_installment;

END CalculateMonthlyInstallment;

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

**Question:** Write a function **HasSufficientBalance** that takes an account ID and an amount as input and returns a boolean indicating whether the account has at least the specified amount.

**CODE**

CREATE OR REPLACE FUNCTION HasSufficientBalance (

    account\_id IN NUMBER,

    amount IN NUMBER

) RETURN BOOLEAN

IS

    current\_balance NUMBER;

BEGIN

    -- Retrieve the current balance of the account

    SELECT balance INTO current\_balance FROM Accounts WHERE account\_id = account\_id;

    -- Check if the balance is sufficient

    IF current\_balance >= amount THEN

        RETURN TRUE;

    ELSE

        RETURN FALSE;

    END IF;

EXCEPTION

    WHEN NO\_DATA\_FOUND THEN

        -- Handle the case where the account ID does not exist

        RETURN FALSE;

    WHEN OTHERS THEN

        -- Handle any other exceptions

        RETURN FALSE;

END HasSufficientBalance;