**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.

**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.

**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.

**Solutions:**

**Scenario 1: Calculate Age**

CREATE OR REPLACE FUNCTION CalculateAge (p\_dob DATE) RETURN NUMBER

IS

v\_age NUMBER;

BEGIN

SELECT FLOOR(MONTHS\_BETWEEN(SYSDATE, p\_dob) / 12) INTO v\_age FROM DUAL;

RETURN v\_age;

END;

**Scenario 2: Calculate Monthly Installment**

CREATE OR REPLACE FUNCTION CalculateMonthlyInstallment (p\_loan\_amount NUMBER, p\_interest\_rate NUMBER, p\_loan\_duration YEARS) RETURN NUMBER

IS

v\_monthly\_installment NUMBER;

BEGIN

v\_monthly\_installment := (p\_loan\_amount \* (1 + (p\_interest\_rate / 100) \* p\_loan\_duration)) / (p\_loan\_duration \* 12);

RETURN v\_monthly\_installment;

END;

**Scenario 3: Has Sufficient Balance**

CREATE OR REPLACE FUNCTION HasSufficientBalance (p\_account\_id NUMBER, p\_amount NUMBER) RETURN BOOLEAN

IS

v\_balance NUMBER;

BEGIN

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

IF v\_balance >= p\_amount THEN

RETURN TRUE;

ELSE

RETURN FALSE;

END IF;END;