**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.
  + **Solution:**

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

BEGIN

RETURN TRUNC(MONTHS\_BETWEEN(SYSDATE, p\_dob) / 12);

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.
  + **Solution:**

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

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

v\_months NUMBER := p\_duration \* 12;

BEGIN

RETURN (p\_loan\_amount \* v\_monthly\_rate) / (1 - POWER(1 + v\_monthly\_rate, -v\_months));

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.
  + **solution:**

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

v\_balance NUMBER;

BEGIN

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

RETURN v\_balance >= p\_amount;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN FALSE;

END HasSufficientBalance;

/