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

CREATE OR REPLACE FUNCTION CalculateAge (

p\_DOB IN DATE

) RETURN NUMBER AS

v\_Age NUMBER;

BEGIN

-- Calculate the age in years

v\_Age := FLOOR(MONTHS\_BETWEEN(SYSDATE, p\_DOB) / 12);

RETURN v\_Age;

END CalculateAge;

/

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

CREATE OR REPLACE FUNCTION CalculateMonthlyInstallment (

p\_LoanAmount IN NUMBER,

p\_InterestRate IN NUMBER,

p\_LoanDurationYears IN NUMBER

) RETURN NUMBER AS

v\_MonthlyRate NUMBER;

v\_NumberOfPayments NUMBER;

v\_MonthlyInstallment NUMBER;

BEGIN

-- Convert annual interest rate to monthly interest rate

v\_MonthlyRate := p\_InterestRate / 12 / 100;

-- Calculate the total number of monthly payments

v\_NumberOfPayments := p\_LoanDurationYears \* 12;

-- Calculate the monthly installment using the formula for an annuity

v\_MonthlyInstallment := p\_LoanAmount \* v\_MonthlyRate /

(1 - POWER(1 + v\_MonthlyRate, -v\_NumberOfPayments));

RETURN v\_MonthlyInstallment;

END CalculateMonthlyInstallment;

/

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

CREATE OR REPLACE FUNCTION HasSufficientBalance (

p\_AccountID IN Accounts.AccountID%TYPE,

p\_Amount IN NUMBER

) RETURN BOOLEAN AS

v\_Balance Accounts.Balance%TYPE;

BEGIN

-- Retrieve the current balance of the account

SELECT Balance INTO v\_Balance

FROM Accounts

WHERE AccountID = p\_AccountID;

-- 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

-- If the account ID does not exist, return FALSE

RETURN FALSE;

END HasSufficientBalance;

/