**Scenario 1:** Calculate the age of customers for eligibility checks. Write a function CalculateAge that takes a customer's date of birth as input and returns their age in years.

**Ans :**

CREATE OR REPLACE FUNCTION CalculateAge (p\_dob DATE)

RETURN NUMBER

IS

v\_age NUMBER;

BEGIN

v\_age := TRUNC(SYSDATE - p\_dob) / 365;

RETURN v\_age;

EXCEPTION

WHEN OTHERS THEN

RETURN -1; -- Indicate an error

END;

**Scenario 2 :** The bank needs to compute the monthly installment for a loan. Write a function **CalculateMonthlyInstallment** that takes the loan amount, interest rate, and loan duration in years as input and returns the monthly installment amount.

**Ans :**

CREATE OR REPLACE FUNCTION CalculateMonthlyInstallment (

p\_loan\_amount NUMBER,

p\_interest\_rate NUMBER,

p\_start\_date DATE,

p\_end\_date DATE

)

RETURN NUMBER

IS

v\_loan\_duration\_years NUMBER;

v\_monthly\_interest\_rate NUMBER;

v\_total\_number\_of\_payments NUMBER;

v\_monthly\_installment NUMBER;

BEGIN

-- Calculate loan duration in years

v\_loan\_duration\_years := MONTHS\_BETWEEN(p\_end\_date, p\_start\_date) / 12;

-- Calculate monthly interest rate

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

-- Calculate total number of payments

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

-- Calculate monthly installment using the formula

v\_monthly\_installment := (p\_loan\_amount \* v\_monthly\_interest\_rate \* POWER(1 + v\_monthly\_interest\_rate, v\_total\_number\_of\_payments))

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

RETURN v\_monthly\_installment;

EXCEPTION

WHEN OTHERS THEN

RETURN -1; -- Indicate an error

END;

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

**Ans :**

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;

RETURN v\_balance >= p\_amount;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN FALSE; -- Account not found

WHEN OTHERS THEN

RETURN FALSE; -- Error occurred

END;