**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

v\_Age NUMBER;

BEGIN

SELECT TRUNC(MONTHS\_BETWEEN(SYSDATE, p\_DOB) / 12) INTO v\_Age

FROM DUAL;

RETURN v\_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.

->**Solution:**

CREATE OR REPLACE FUNCTION CalculateMonthlyInstallment (

p\_LoanAmount IN NUMBER,

p\_InterestRate IN NUMBER,

p\_LoanDurationYears IN NUMBER

) RETURN NUMBER

IS

v\_MonthlyRate NUMBER;

v\_NumberOfPayments NUMBER;

v\_MonthlyInstallment NUMBER;

BEGIN

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

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

IF v\_MonthlyRate = 0 THEN

v\_MonthlyInstallment := p\_LoanAmount / v\_NumberOfPayments;

ELSE

v\_MonthlyInstallment := p\_LoanAmount \* v\_MonthlyRate / (1 - POWER(1 + v\_MonthlyRate, -v\_NumberOfPayments));

END IF;

RETURN v\_MonthlyInstallment;

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\_AccountID IN NUMBER,

p\_Amount IN NUMBER

) RETURN BOOLEAN

IS

v\_Balance NUMBER;

BEGIN

SELECT Balance INTO v\_Balance

FROM Accounts

WHERE AccountID = p\_AccountID;

RETURN v\_Balance >= p\_Amount;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN FALSE;

END HasSufficientBalance;

/