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.

CREATE OR REPLACE FUNCTION CalculateAge(dob DATE) RETURN NUMBER IS

age NUMBER;

BEGIN

age := FLOOR(MONTHS\_BETWEEN(SYSDATE, dob) / 12);

RETURN 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.

CREATE OR REPLACE FUNCTION CalculateMonthlyInstallment(

loanAmount NUMBER,

annualInterestRate NUMBER,

loanDurationYears NUMBER

) RETURN NUMBER IS

monthlyInstallment NUMBER;

monthlyInterestRate NUMBER;

totalMonths NUMBER;

BEGIN

monthlyInterestRate := annualInterestRate / 1200;

totalMonths := loanDurationYears \* 12;

IF monthlyInterestRate = 0 THEN

monthlyInstallment := loanAmount / totalMonths;

ELSE

monthlyInstallment := loanAmount \* monthlyInterestRate \* POWER((1 + monthlyInterestRate), totalMonths)

/ (POWER((1 + monthlyInterestRate), totalMonths) - 1);

END IF;

RETURN 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.

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;

IF v\_Balance >= p\_Amount THEN

RETURN TRUE;

ELSE

RETURN FALSE;

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN FALSE;

WHEN OTHERS THEN

RAISE;

END HasSufficientBalance;

/