**Functions**

**Scenario 1: Calculate Age of Customers**

**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(

p\_dob DATE

) RETURN NUMBER IS

v\_age NUMBER;

BEGIN

v\_age := FLOOR(MONTHS\_BETWEEN(SYSDATE, p\_dob) / 12);

RETURN v\_age;

END;  
  
**Scenario 2: Compute 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(

p\_loanAmount NUMBER,

p\_interestRate NUMBER,

p\_durationYears NUMBER

) RETURN NUMBER IS

v\_monthlyRate NUMBER;

v\_totalPayments NUMBER;

v\_monthlyInstallment NUMBER;

BEGIN

v\_monthlyRate := p\_interestRate / 100 / 12;

v\_totalPayments := p\_durationYears \* 12;

v\_monthlyInstallment := p\_loanAmount \* v\_monthlyRate / (1 - POWER(1 + v\_monthlyRate, -v\_totalPayments));

RETURN v\_monthlyInstallment;

END;  
  
**Scenario 3: Check Sufficient Balance**

**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 NUMBER,

p\_amount NUMBER

) RETURN BOOLEAN IS

v\_balance Accounts.Balance%TYPE;

BEGIN

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

RETURN v\_balance >= p\_amount;

END;