**4) 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.**

**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.**

**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.**

Scenario1-

CREATE OR REPLACE FUNCTION CalculateAge (

p\_birth\_date IN DATE

) RETURN NUMBER IS

v\_age NUMBER;

BEGIN

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

RETURN v\_age;

END;

/

ii)Scenario 2-

CREATE OR REPLACE FUNCTION CalculateMonthlyInstallment (

p\_loan\_amount IN NUMBER,

p\_interest\_rate IN NUMBER,

p\_duration\_years IN NUMBER

) RETURN NUMBER IS

v\_monthly\_rate NUMBER;

v\_num\_payments NUMBER;

v\_monthly\_installment NUMBER;

BEGIN

v\_monthly\_rate := p\_interest\_rate / 1200;

v\_num\_payments := p\_duration\_years \* 12;

v\_monthly\_installment := (p\_loan\_amount \* v\_monthly\_rate) / (1 - POWER(1 + v\_monthly\_rate, -v\_num\_payments));

RETURN v\_monthly\_installment;

END;

/

iii)Scenario 3-

CREATE OR REPLACE FUNCTION HasSufficientBalance (

p\_account\_id IN NUMBER,

p\_amount IN NUMBER

) RETURN BOOLEAN IS

v\_balance NUMBER;

BEGIN

SELECT balance INTO v\_balance FROM accounts WHERE account\_id = p\_account\_id;

IF v\_balance >= p\_amount THEN

RETURN TRUE;

ELSE

RETURN FALSE;

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN FALSE;

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error: An unexpected error occurred while checking the balance.');

RETURN FALSE;

END;

/