**Exercise 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.

**Code:** **Scenario 1**

CREATE OR REPLACE FUNCTION CalculateAge( cust\_dob IN DATE)

Return number

IS

cust\_age NUMBER;

Begin

cust\_age:=(SYSDATE-cust\_dob)/365;

RETURN cust\_age;

END;

/

SET SERVEROUT ON;

DECLARE

age NUMBER;

BEGIN

age := CalculateAge(TO\_DATE('2002-08-26', 'YYYY-MM-DD'));

DBMS\_OUTPUT.PUT\_LINE('Age of Customer: ' || TRUNC(age));

END;

/

**Code:** **Scenario 2**

CREATE OR REPLACE FUNCTION CalculateMonthlyInstallment (

loan\_amount NUMBER,

interest\_rate NUMBER,

loan\_duration NUMBER

)

RETURN NUMBER

IS

monthly\_interest NUMBER;

monthly\_install NUMBER;

BEGIN

monthly\_interest := interest\_rate / 1200;

monthly\_install := (loan\_amount\*monthly\_interest\*POWER(1+monthly\_interest,loan\_duration\*12)) / (POWER(1+monthly\_interest,loan\_duration \* 12) - 1);

RETURN monthly\_install;

END;

/

SQL> SET SERVEROUTPUT ON;

SQL> DECLARE

res NUMBER;

BEGIN

res := CalculateMonthlyInstallment(100,4,2);

DBMS\_OUTPUT.PUT\_LINE(TRUNC(res));

END;

/

PL/SQL procedure successfully completed.

**Code:** **Scenario 3**

CREATE OR REPLACE FUNCTION HasSufficientBalance (

account\_id IN NUMBER,

amount IN NUMBER

)

RETURN BOOLEAN

IS

f\_Balance NUMBER;

BEGIN

SELECT balance INTO f\_balance FROM ACCOUNTS where accountid=account\_id;

IF f\_balance >= amount THEN

RETURN TRUE;

ELSE

RETURN FALSE;

END IF;

END;

/

SQL> SET SERVEROUTPUT ON;

SQL> DECLARE

ans BOOLEAN;

BEGIN

ans := HasSufficientBalance(1, 1000);

IF ans THEN

DBMS\_OUTPUT.PUT\_LINE('TRUE');

ELSE

DBMS\_OUTPUT.PUT\_LINE('FALSE');

END IF;

END;

/

TRUE

PL/SQL procedure successfully completed.