**EXERCISE 4**-**FUNCTIONS**

**SCENARIO 1: CalculateAge function**

**CREATING PROCEDURE:**

CREATE OR REPLACE FUNCTION CalculateAge (

    p\_dob DATE

) RETURN NUMBER

IS

    v\_age NUMBER;

BEGIN

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

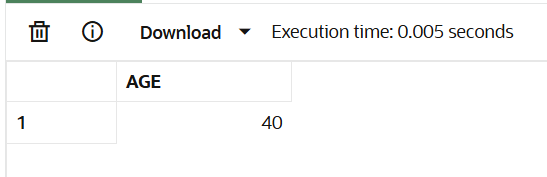
    RETURN v\_age;

END;

/

SELECT CalculateAge(TO\_DATE('1985-05-15','YYYY-MM-DD')) AS Age FROM dual;

**OUTPUT:**

****

**SCENARIO 2:** **CalculateMonthlyInstallment function**

CREATE OR REPLACE FUNCTION CalculateMonthlyInstallment (

    p\_loan\_amount NUMBER,

    p\_annual\_interest\_rate NUMBER,

    p\_years NUMBER

) RETURN NUMBER

IS

    v\_monthly\_rate NUMBER;

    v\_n NUMBER;

    v\_emi NUMBER;

BEGIN

    v\_monthly\_rate := p\_annual\_interest\_rate / (12\*100);

    v\_n := p\_years \* 12;

    v\_emi := p\_loan\_amount \* v\_monthly\_rate \* POWER(1+v\_monthly\_rate, v\_n) /

            (POWER(1+v\_monthly\_rate, v\_n) - 1);

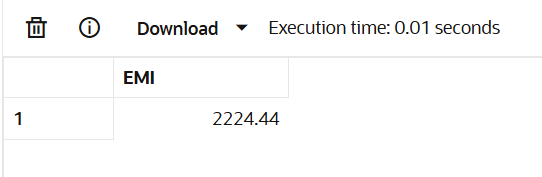
    RETURN ROUND(v\_emi,2);

END;

/

SELECT CalculateMonthlyInstallment(100000, 12, 5) AS EMI FROM dual;

**OUTPUT:**

****

**SCENARIO 3:** **HasSufficientBalance function**

CREATE OR REPLACE FUNCTION HasSufficientBalance (

    p\_account\_id NUMBER,

    p\_amount NUMBER

) RETURN BOOLEAN

IS

    v\_balance NUMBER;

BEGIN

    SELECT Balance INTO v\_balance

    FROM Accounts

    WHERE AccountID = p\_account\_id;

    IF v\_balance >= p\_amount THEN

        RETURN TRUE;

    ELSE

        RETURN FALSE;

    END IF;

EXCEPTION

    WHEN NO\_DATA\_FOUND THEN

        RETURN FALSE;

END;

/

SET SERVEROUTPUT ON;

DECLARE

    v\_result BOOLEAN;

BEGIN

    v\_result := HasSufficientBalance(101, 1000);  -- check for 1000

    IF v\_result THEN

        DBMS\_OUTPUT.PUT\_LINE('Sufficient balance.');

    ELSE

        DBMS\_OUTPUT.PUT\_LINE('Insufficient balance.');

    END IF;

END;

/

**OUTPUT:**

**:**