Exercise 4

-- Scenario 1  
CREATE OR REPLACE FUNCTION CalculateAge(p\_dob DATE) RETURN NUMBER AS  
BEGIN  
 RETURN TRUNC(MONTHS\_BETWEEN(SYSDATE, p\_dob) / 12);  
END;  
/  
  
-- Scenario 2  
CREATE OR REPLACE FUNCTION CalculateMonthlyInstallment(p\_amount NUMBER, p\_rate NUMBER, p\_years NUMBER) RETURN NUMBER AS  
 v\_months NUMBER := p\_years \* 12;  
 v\_monthly\_rate NUMBER := p\_rate / (12 \* 100);  
BEGIN  
 RETURN (p\_amount \* v\_monthly\_rate) / (1 - POWER(1 + v\_monthly\_rate, -v\_months));  
END;  
/  
  
-- Scenario 3  
CREATE OR REPLACE FUNCTION HasSufficientBalance(p\_accountid NUMBER, p\_amount NUMBER) RETURN BOOLEAN AS  
 v\_balance NUMBER;  
BEGIN  
 SELECT Balance INTO v\_balance FROM Accounts WHERE AccountID = p\_accountid;  
 RETURN v\_balance >= p\_amount;  
END;  
/  
  
OUTPUT:

-- Input: Date of birth = 1990-01-01SELECT CalculateAge(DATE '1990-01-01') AS Age FROM DUAL;

-- Output:-- AGE 35

-- Input: Amount = 10000, Annual Rate = 10%, Years = 2SELECT CalculateMonthlyInstallment(10000, 10, 2) AS Installment FROM DUAL;

-- Output:-- INSTALLMENT 461.45

INSERT INTO Accounts (AccountID, Balance) VALUES (101, 1000);

DECLARE

res BOOLEAN;

BEGIN

res := HasSufficientBalance(101, 500);

IF res THEN DBMS\_OUTPUT.PUT\_LINE('TRUE'); ELSE DBMS\_OUTPUT.PUT\_LINE('FALSE'); END IF;

res := HasSufficientBalance(101, 1500);

IF res THEN DBMS\_OUTPUT.PUT\_LINE('TRUE'); ELSE DBMS\_OUTPUT.PUT\_LINE('FALSE'); END IF;

END;

/

Output:

TRUE

FALSE