**Exercise 4: Functions**

CREATE OR REPLACE FUNCTION CalculateAge(p\_dob DATE)

RETURN NUMBER

IS

  v\_age NUMBER;

BEGIN

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

  RETURN v\_age;

END;

/

SELECT Name, DOB, CalculateAge(DOB) AS Age

FROM Customers;

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\_months NUMBER;

  v\_emi NUMBER;

BEGIN

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

  v\_months := p\_years \* 12;

  IF v\_monthly\_rate = 0 THEN

    v\_emi := p\_loan\_amount / v\_months;

  ELSE

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

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

  END IF;

  RETURN ROUND(v\_emi, 2);

END;

/

SELECT CalculateMonthlyInstallment(50000, 7.5, 5) AS EMI FROM dual;

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;

  RETURN v\_balance >= p\_amount;

EXCEPTION

  WHEN NO\_DATA\_FOUND THEN

    RETURN FALSE;

END;

/

DECLARE

  result BOOLEAN;

BEGIN

  result := HasSufficientBalance(1, 500);

  IF result THEN

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

  ELSE

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

  END IF;

END;

/











