Week-2

PL SQL programming

**Exercise 4: Functions**

---------------------------------------

-- CLEANUP

---------------------------------------

BEGIN

EXECUTE IMMEDIATE 'DROP TABLE Accounts';

EXCEPTION

WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE; END IF;

END;

/

---------------------------------------

-- CREATE TABLES & SAMPLE DATA

---------------------------------------

CREATE TABLE Accounts (

AccountID NUMBER PRIMARY KEY,

CustomerID NUMBER,

AccountType VARCHAR2(20),

Balance NUMBER,

LastModified DATE

);

INSERT INTO Accounts VALUES (1, 1, 'Savings', 5000, SYSDATE);

INSERT INTO Accounts VALUES (2, 2, 'Checking', 2000, SYSDATE);

COMMIT;

---------------------------------------

-- ENABLE OUTPUT

---------------------------------------

SET SERVEROUTPUT ON

---------------------------------------

-- Scenario 1: Calculate the age of customers

---------------------------------------

CREATE OR REPLACE FUNCTION CalculateAge(dob DATE) RETURN NUMBER IS

age NUMBER;

BEGIN

age := TRUNC(MONTHS\_BETWEEN(SYSDATE, dob) / 12);

RETURN age;

END;

/

-- Example call with output

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Age is: '||CalculateAge(TO\_DATE('1990-07-20','YYYY-MM-DD')));

END;

/

---------------------------------------

-- Scenario 2: Calculate monthly installment

---------------------------------------

CREATE OR REPLACE FUNCTION CalculateMonthlyInstallment(

loan\_amount NUMBER,

annual\_interest\_rate NUMBER,

loan\_years NUMBER

) RETURN NUMBER IS

r NUMBER;

n NUMBER;

emi NUMBER;

BEGIN

r := annual\_interest\_rate / 12 / 100; -- monthly rate

n := loan\_years \* 12;

emi := loan\_amount \* r \* POWER(1 + r, n) / (POWER(1 + r, n) - 1);

RETURN emi;

END;

/

-- Example call with output

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Monthly Installment: '||

ROUND(CalculateMonthlyInstallment(10000, 6, 2),2));

END;

/

---------------------------------------

-- Scenario 3: Check if customer has sufficient balance

---------------------------------------

CREATE OR REPLACE FUNCTION HasSufficientBalance(

acc\_id NUMBER,

amount NUMBER

) RETURN VARCHAR2 IS

acc\_balance NUMBER;

BEGIN

SELECT Balance INTO acc\_balance FROM Accounts WHERE AccountID = acc\_id;

IF acc\_balance >= amount THEN

RETURN 'TRUE';

ELSE

RETURN 'FALSE';

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 'FALSE';

END;

/

-- Example call with output

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Account 1 has sufficient balance for 3000? '||

HasSufficientBalance(1,3000));

DBMS\_OUTPUT.PUT\_LINE('Account 2 has sufficient balance for 3000? '||

HasSufficientBalance(2,3000));

END;

/

