DO $$

BEGIN

IF EXISTS (

SELECT FROM information\_schema.tables

WHERE table\_name = 'accounts'

) THEN

EXECUTE 'DROP TABLE accounts';

END IF;

END

$$;

CREATE TABLE Accounts (

AccountID INTEGER PRIMARY KEY,

CustomerID INTEGER,

AccountType VARCHAR(20),

Balance NUMERIC,

LastModified DATE

);

INSERT INTO Accounts VALUES

(1, 101, 'Savings', 5000, CURRENT\_DATE),

(2, 102, 'Checking', 12000, CURRENT\_DATE);

CREATE OR REPLACE FUNCTION CalculateAge(p\_dob DATE)

RETURNS INTEGER AS $$

DECLARE

v\_age INTEGER;

BEGIN

v\_age := DATE\_PART('year', AGE(CURRENT\_DATE, p\_dob));

RETURN v\_age;

END;

$$ LANGUAGE plpgsql;

CREATE OR REPLACE FUNCTION CalculateMonthlyInstallment(

p\_loanAmount NUMERIC,

p\_interestRate NUMERIC,

p\_durationYears NUMERIC

)

RETURNS NUMERIC AS $$

DECLARE

v\_monthlyRate NUMERIC;

v\_durationMonths NUMERIC;

v\_payment NUMERIC;

BEGIN

v\_monthlyRate := (p\_interestRate / 100) / 12;

v\_durationMonths := p\_durationYears \* 12;

IF v\_monthlyRate = 0 THEN

v\_payment := p\_loanAmount / v\_durationMonths;

ELSE

v\_payment := (p\_loanAmount \* v\_monthlyRate) / (1 - POWER(1 + v\_monthlyRate, -v\_durationMonths));

END IF;

RETURN ROUND(v\_payment, 2);

END;

$$ LANGUAGE plpgsql;

CREATE OR REPLACE FUNCTION HasSufficientBalance(

p\_accountId INT,

p\_amount NUMERIC

)

RETURNS BOOLEAN AS $$

DECLARE

v\_currentBalance NUMERIC;

BEGIN

SELECT Balance INTO v\_currentBalance

FROM Accounts

WHERE AccountID = p\_accountId;

IF NOT FOUND THEN

RETURN FALSE;

END IF;

RETURN v\_currentBalance >= p\_amount;

END;

$$ LANGUAGE plpgsql;

SELECT 'Customer DOB 1990-05-10 Age:' AS Label, CalculateAge(DATE '1990-05-10') AS Age;

SELECT 'Monthly Installment on 50000 at 6% over 5 years:' AS Label,

CalculateMonthlyInstallment(50000, 6, 5) AS Installment;

SELECT 'Account 1 has at least 3000?' AS Label, HasSufficientBalance(1, 3000) AS Result;

SELECT 'Account 2 has at least 15000?' AS Label, HasSufficientBalance(2, 15000) AS Result;