PL/SQL Exercises - Complete Solutions

# Exercise 1: Control Structures

## 1.1 Apply Discount to Senior Customers

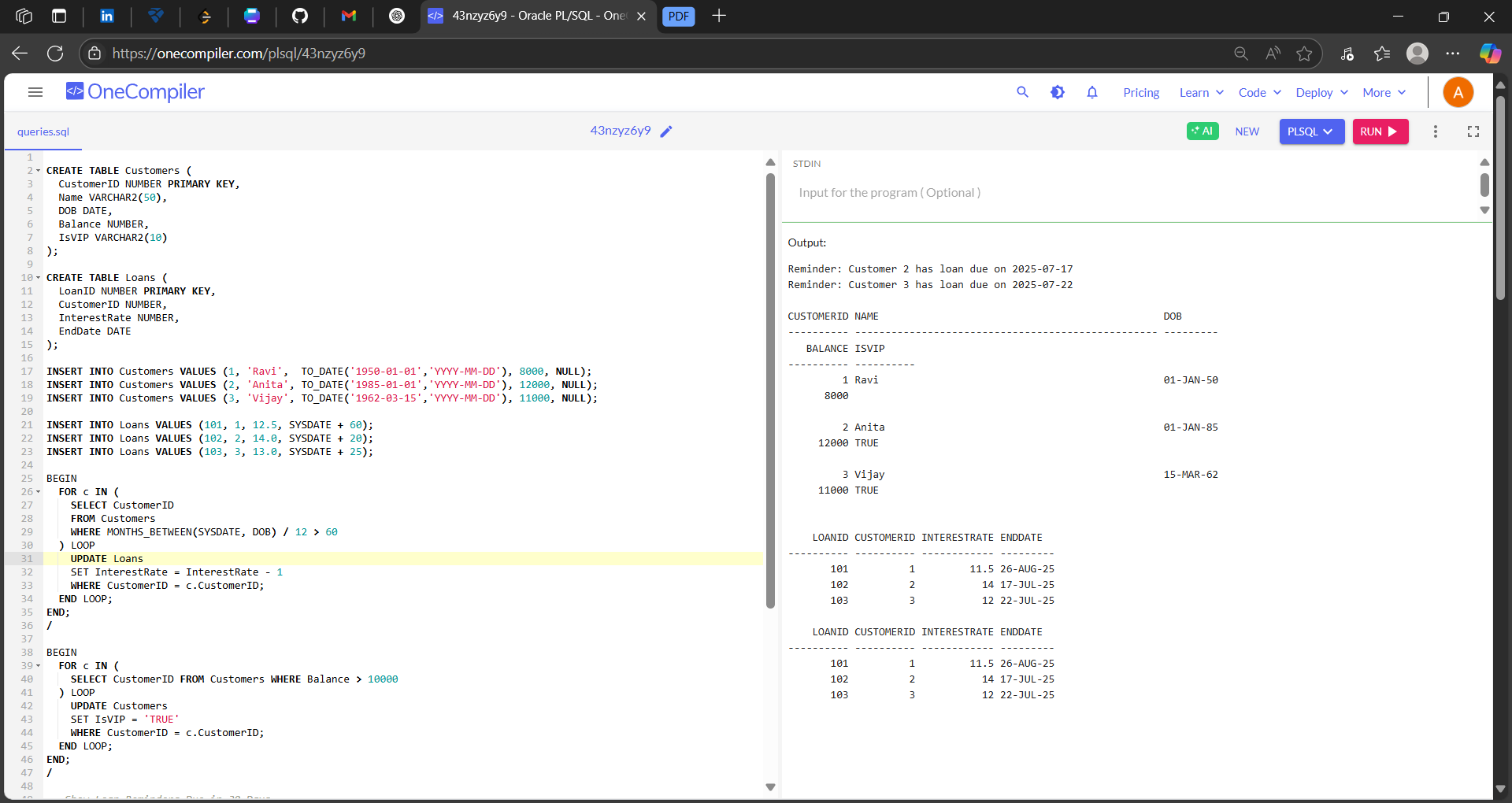
BEGIN  
 FOR c IN (SELECT CustomerID FROM Customers WHERE MONTHS\_BETWEEN(SYSDATE, DOB)/12 > 60) LOOP  
 UPDATE Loans  
 SET InterestRate = InterestRate - 1  
 WHERE CustomerID = c.CustomerID;  
 END LOOP;  
END;

## 1.2 Promote to VIP Status

BEGIN  
 FOR c IN (SELECT CustomerID FROM Customers WHERE Balance > 10000) LOOP  
 UPDATE Customers  
 SET IsVIP = 'TRUE'  
 WHERE CustomerID = c.CustomerID;  
 END LOOP;  
END;

## 1.3 Loan Due Reminder

BEGIN  
 FOR l IN (  
 SELECT LoanID, CustomerID, EndDate FROM Loans  
 WHERE EndDate <= SYSDATE + 30  
 ) LOOP  
 DBMS\_OUTPUT.PUT\_LINE('Reminder: Customer ' || l.CustomerID || ' has loan due on ' || TO\_CHAR(l.EndDate));  
 END LOOP;  
END;



# Exercise 3: Stored Procedures

## 3.1 ProcessMonthlyInterest

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS  
BEGIN  
 UPDATE Accounts  
 SET Balance = Balance \* 1.01  
 WHERE AccountType = 'Savings';  
END;

## 3.2 UpdateEmployeeBonus

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(p\_dept VARCHAR2, p\_bonus NUMBER) IS  
BEGIN  
 UPDATE Employees  
 SET Salary = Salary + (Salary \* p\_bonus / 100)  
 WHERE Department = p\_dept;  
END;

## 3.3 TransferFunds

CREATE OR REPLACE PROCEDURE TransferFunds(p\_from NUMBER, p\_to NUMBER, p\_amt NUMBER) IS  
 v\_balance NUMBER;  
BEGIN  
 SELECT Balance INTO v\_balance FROM Accounts WHERE AccountID = p\_from;  
  
 IF v\_balance >= p\_amt THEN  
 UPDATE Accounts SET Balance = Balance - p\_amt WHERE AccountID = p\_from;  
 UPDATE Accounts SET Balance = Balance + p\_amt WHERE AccountID = p\_to;  
 ELSE  
 DBMS\_OUTPUT.PUT\_LINE('Insufficient balance.');  
 END IF;  
END;

