**Deep Skilling - Week 2 Mandatory Hands-on Exercises**

**Skill : PL/SQL Programming**

**TOOL USED: onecompiler.com/plsql**

**Exercise - 1: Control Structures**

**Code:**

**CREATE TABLE Customers (**

**CustomerID NUMBER PRIMARY KEY,**

**Name VARCHAR2(100),**

**DOB DATE,**

**Balance NUMBER,**

**LastModified DATE**

**);**

**CREATE TABLE Loans (**

**LoanID NUMBER PRIMARY KEY,**

**CustomerID NUMBER,**

**LoanAmount NUMBER,**

**InterestRate NUMBER,**

**StartDate DATE,**

**EndDate DATE,**

**FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)**

**);**

**BEGIN**

**INSERT INTO Customers VALUES (1011, 'Deepikaa', TO\_DATE('1985-03-15', 'YYYY-MM-DD'), 31000, SYSDATE);**

**INSERT INTO Customers VALUES (1021, 'Pramilaa', TO\_DATE('2005-05-30', 'YYYY-MM-DD'), 11000, SYSDATE);**

**INSERT INTO Customers VALUES (1031, 'Rishuti', TO\_DATE('1947-06-26', 'YYYY-MM-DD'), 1000, SYSDATE);**

**INSERT INTO Customers VALUES (1041, 'Lalithya', TO\_DATE('1950-07-01', 'YYYY-MM-DD'), 1000, SYSDATE);**

**INSERT INTO Customers VALUES (1051, 'Rishithaa', TO\_DATE('1949-08-05', 'YYYY-MM-DD'), 21000, SYSDATE);**

**INSERT INTO Customers VALUES (1061, 'Pranitha', TO\_DATE('2015-09-03', 'YYYY-MM-DD'), 1000, SYSDATE);**

**END;**

**/**

**BEGIN**

**INSERT INTO Loans VALUES (1101, 1011, 5000, 5, SYSDATE, SYSDATE + 20);**

**INSERT INTO Loans VALUES (1102, 1021, 5000, 5, SYSDATE, SYSDATE + 73);**

**INSERT INTO Loans VALUES (1103, 1031, 5000, 5, SYSDATE, SYSDATE + 30);**

**INSERT INTO Loans VALUES (1104, 1041, 5000, 5, SYSDATE, SYSDATE + 67);**

**INSERT INTO Loans VALUES (1105, 1051, 5000, 5, SYSDATE, SYSDATE + 80);**

**INSERT INTO Loans VALUES (1106, 1061, 5000, 5, SYSDATE, SYSDATE + 25);**

**END;**

**/**

**-- Scenario-1 : Applying Discount to Loan Interest to Customers above age 60**

**BEGIN**

**FOR record IN (SELECT c.CustomerID, l.LoanID, l.InterestRate,**

**TRUNC(MONTHS\_BETWEEN(SYSDATE, c.DOB)/12) AS age**

**FROM Customers c**

**JOIN Loans l ON c.CustomerID = l.CustomerID)**

**LOOP**

**IF record.age > 60 THEN**

**UPDATE Loans**

**SET InterestRate = InterestRate - 1**

**WHERE LoanID = record.LoanID;**

**DBMS\_OUTPUT.PUT\_LINE('InterestRate updated for CustomerID: ' || record.CustomerID || ' whose age is: ' || record.age);**

**END IF;**

**END LOOP;**

**END;**

**/**

**-- Scenario-2 : Promoting Customers to VIP based on Balance**

**ALTER TABLE Customers ADD IsVIP VARCHAR2(10);**

**BEGIN**

**FOR record IN (SELECT CustomerID, Balance FROM Customers)**

**LOOP**

**IF record.Balance > 10000 THEN**

**UPDATE Customers**

**SET IsVIP = 'TRUE'**

**WHERE CustomerID = record.CustomerID;**

**DBMS\_OUTPUT.PUT\_LINE(record.CustomerID || ' is now a VIP');**

**END IF;**

**END LOOP;**

**END;**

**/**

**-- Scenario-3 : Sending reminder to Customers whose loans are due in 30 days**

**BEGIN**

**FOR record IN (**

**SELECT c.Name, l.EndDate**

**FROM Customers c**

**JOIN Loans l ON c.CustomerID = l.CustomerID**

**WHERE l.EndDate <= SYSDATE + 30**

**)**

**LOOP**

**DBMS\_OUTPUT.PUT\_LINE('Reminder: Loan for ' || record.Name || ' is due on ' || TO\_CHAR(record.EndDate, 'DD-MON-YYYY'));**

**END LOOP;**

**END;**

**/**

**Output:**

**InterestRate updated for CustomerID: 1031 whose age is: 78**

**InterestRate updated for CustomerID: 1041 whose age is: 74**

**InterestRate updated for CustomerID: 1051 whose age is: 75**

**1011 is now a VIP**

**1021 is now a VIP**

**1051 is now a VIP**

**Reminder: Loan for Deepikaa is due on 18-JUL-2025**

**Reminder: Loan for Rishuti is due on 28-JUL-2025**

**Reminder: Loan for Pranitha is due on 23-JUL-2025**

**Exercise - 3: Stored Procedures**

**Code:**

**CREATE TABLE Customers (**

**CustomerID NUMBER PRIMARY KEY,**

**Name VARCHAR2(100),**

**DOB DATE,**

**Balance NUMBER,**

**LastModified DATE**

**);**

**CREATE TABLE Accounts (**

**AccountID NUMBER PRIMARY KEY,**

**CustomerID NUMBER,**

**AccountType VARCHAR2(20),**

**Balance NUMBER,**

**LastModified DATE,**

**FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)**

**);**

**CREATE TABLE Employees (**

**EmployeeID NUMBER PRIMARY KEY,**

**Name VARCHAR2(100),**

**Position VARCHAR2(50),**

**Salary NUMBER,**

**Department VARCHAR2(50),**

**HireDate DATE**

**);**

**BEGIN**

**INSERT INTO Customers VALUES (1011, 'Deepikaa', TO\_DATE('1985-03-15', 'YYYY-MM-DD'), 31000, SYSDATE);**

**INSERT INTO Customers VALUES (1021, 'Pramilaa', TO\_DATE('2005-05-30', 'YYYY-MM-DD'), 11000, SYSDATE);**

**INSERT INTO Customers VALUES (1031, 'Rishuti', TO\_DATE('1947-06-26', 'YYYY-MM-DD'), 1000, SYSDATE);**

**INSERT INTO Customers VALUES (1041, 'Lalithya', TO\_DATE('1950-07-01', 'YYYY-MM-DD'), 1000, SYSDATE);**

**INSERT INTO Customers VALUES (1051, 'Rishithaa', TO\_DATE('1949-08-05', 'YYYY-MM-DD'), 21000, SYSDATE);**

**INSERT INTO Customers VALUES (1061, 'Pranitha', TO\_DATE('2015-09-03', 'YYYY-MM-DD'), 1000, SYSDATE);**

**END;**

**/**

**BEGIN**

**INSERT INTO Accounts VALUES (11014, 1011, 'Savings', 11000, SYSDATE);**

**INSERT INTO Accounts VALUES (11015, 1021, 'Current', 10000, SYSDATE);**

**INSERT INTO Accounts VALUES (11016, 1031, 'Savings', 21000, SYSDATE);**

**INSERT INTO Accounts VALUES (11017, 1041, 'Savings', 9000, SYSDATE);**

**INSERT INTO Accounts VALUES (11018, 1051, 'Current', 1000, SYSDATE);**

**INSERT INTO Accounts VALUES (11019, 1061, 'Savings', 6000, SYSDATE);**

**END;**

**/**

**BEGIN**

**INSERT INTO Employees VALUES (4501, 'Millie', 'Manager', 70000, 'HR', TO\_DATE('2005-07-15', 'YYYY-MM-DD'));**

**INSERT INTO Employees VALUES (4502, 'Bobby', 'Intern', 70000, 'Dev', TO\_DATE('2025-06-20', 'YYYY-MM-DD'));**

**INSERT INTO Employees VALUES (4503, 'Brown', 'Intern', 70000, 'Cloud Service', TO\_DATE('2025-02-15', 'YYYY-MM-DD'));**

**INSERT INTO Employees VALUES (4504, 'Stevee', 'SOC Analyst', 70000, 'Cybersecurity', TO\_DATE('2021-01-29', 'YYYY-MM-DD'));**

**INSERT INTO Employees VALUES (4505, 'Maxwell', 'Manager', 70000, 'HR', TO\_DATE('2018-01-31', 'YYYY-MM-DD'));**

**END;**

**/**

**-- Scenario-1 : Processing Monthly Interest for all saving Accounts**

**CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS**

**BEGIN**

**FOR record IN (SELECT AccountID, Balance FROM Accounts WHERE AccountType = 'Savings')**

**LOOP**

**UPDATE Accounts**

**SET Balance = Balance + (Balance \* 0.01), LastModified = SYSDATE**

**WHERE AccountID = record.AccountID;**

**DBMS\_OUTPUT.PUT\_LINE('Updated balance for AccountID = ' || record.AccountID);**

**END LOOP;**

**END;**

**/**

**BEGIN**

**ProcessMonthlyInterest;**

**END;**

**/**

**--Scenario-2 : Implementing bonus for the employees based on performance**

**CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (dept IN VARCHAR2, bonusPoints IN integer) IS**

**BEGIN**

**FOR record IN (SELECT EmployeeID, Department, Salary FROM Employees WHERE Department = dept)**

**LOOP**

**UPDATE Employees**

**SET Salary = Salary + (Salary \* bonusPoints)**

**WHERE record.EmployeeID = EmployeeID;**

**DBMS\_OUTPUT.PUT\_LINE('Added Bonus points to employee : ' || record.EmployeeID);**

**END LOOP;**

**END;**

**/**

**BEGIN**

**UpdateEmployeeBonus('Cybersecurity', 0.09);**

**END;**

**/**

**--Scenario-3 : Transferring funds between Customers Accounts**

**CREATE OR REPLACE PROCEDURE TransferFunds (**

**from\_account IN NUMBER,**

**to\_account IN NUMBER,**

**fund IN NUMBER) IS**

**from\_balance NUMBER;**

**BEGIN**

**SELECT Balance INTO from\_balance FROM Accounts WHERE AccountID = from\_account FOR UPDATE;**

**IF from\_balance < fund THEN**

**RAISE\_APPLICATION\_ERROR(-20001, 'Insufficient Balance in the Account.');**

**END IF;**

**UPDATE Accounts**

**SET Balance = Balance - fund, LastModified = SYSDATE**

**WHERE AccountID = from\_account;**

**UPDATE Accounts**

**SET Balance = Balance + fund, LastModified = SYSDATE**

**WHERE AccountID = to\_account;**

**DBMS\_OUTPUT.PUT\_LINE('Transfered Funds successfully from ' || from\_account || ' to account ' || to\_account || ' amount ' || fund);**

**END;**

**/**

**BEGIN**

**TransferFunds(11015, 11017, 1000);**

**TransferFunds(11017, 11018, 3000);**

**TransferFunds(11018, 11014, 6000);**

**END;**

**/**

**Output:**

**Updated balance for AccountID = 11014**

**Updated balance for AccountID = 11016**

**Updated balance for AccountID = 11017**

**Updated balance for AccountID = 11019**

**Added Bonus points to employee : 4504**

**Transferred Funds successfully from 11015 to account 11017 amount 1000**

**Transferred Funds successfully from 11017 to account 11018 amount 3000**

**BEGIN**

**\***

**ERROR at line 1:**

**ORA-20001: Insufficient Balance in the Account.**

**ORA-06512: at "C##43NZJPPFY\_43P4ZMXHV.TRANSFERFUNDS", line 10**

**ORA-06512: at line 4**