**PL/SQL Hands-On**

**Exercise 1: Control Structures**

**Code:**

**Customers Table**

CREATE TABLE Customers (  
 CustomerID NUMBER PRIMARY KEY,  
 Name VARCHAR2(100),  
 Age NUMBER,  
 Balance NUMBER,  
 IsVIP CHAR(1) DEFAULT 'N'  
);

**Loans Table**

CREATE TABLE Loans (  
 LoanID NUMBER PRIMARY KEY,  
 CustomerID NUMBER,  
 InterestRate NUMBER,  
 DueDate DATE  
);

**Sample Data insertion**

-- Insert Customers  
INSERT INTO Customers VALUES (1, 'John Doe', 65, 12000, 'N');  
INSERT INTO Customers VALUES (2, 'Alice Smith', 45, 8000, 'N');  
INSERT INTO Customers VALUES (3, 'Bob Johnson', 70, 5000, 'N');  
  
-- Insert Loans  
INSERT INTO Loans VALUES (101, 1, 10.0, SYSDATE + 10);  
INSERT INTO Loans VALUES (102, 2, 9.5, SYSDATE + 40);  
INSERT INTO Loans VALUES (103, 3, 8.0, SYSDATE + 5);  
  
COMMIT;

**Scenario 1 - Interest Discount for Customers Over 60**

BEGIN  
 FOR rec IN (SELECT CustomerID FROM Customers WHERE Age > 60) LOOP  
 UPDATE Loans  
 SET InterestRate = InterestRate - (InterestRate \* 0.01)  
 WHERE CustomerID = rec.CustomerID;  
 END LOOP;  
 COMMIT;  
END;  
/

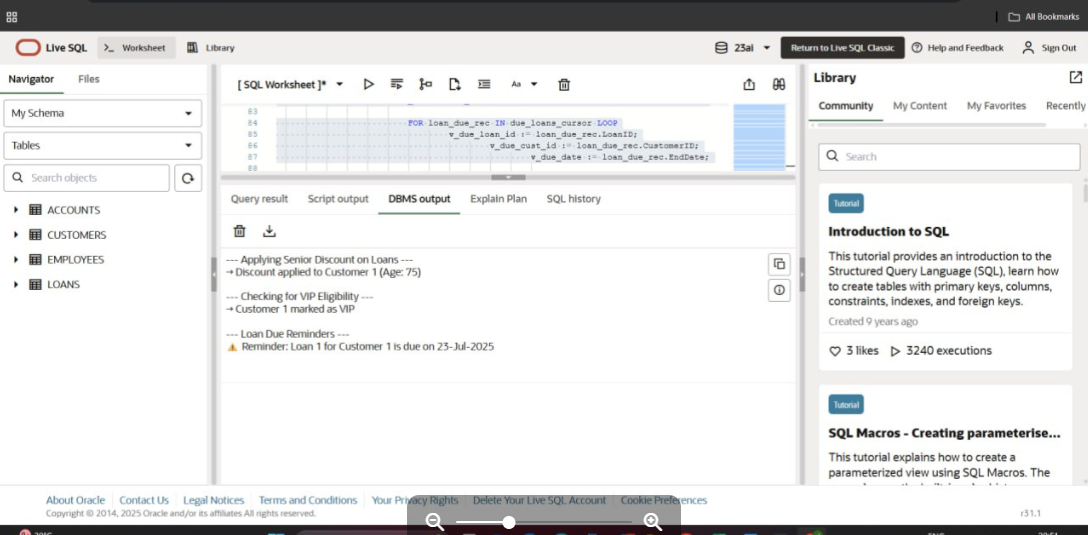
**Scenario 2 - Promote VIP Customers**

BEGIN  
 FOR rec IN (SELECT CustomerID FROM Customers WHERE Balance > 10000) LOOP  
 UPDATE Customers  
 SET IsVIP = 'Y'  
 WHERE CustomerID = rec.CustomerID;  
 END LOOP;  
 COMMIT;  
END;  
/

**Scenario 3 - Loan Due Reminders**

SELECT c.Name, l.DueDate  
FROM Loans l  
JOIN Customers c ON l.CustomerID = c.CustomerID  
WHERE l.DueDate <= SYSDATE + 30;

Output :



**Exercise 3**

**Scenario 1: Process Monthly Interest for Savings Accounts**

***Stored Procedure: ProcessMonthlyInterest***

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest AS  
BEGIN  
 UPDATE Accounts  
 SET Balance = Balance + (Balance \* 0.01)  
 WHERE AccountType = 'SAVINGS';  
 COMMIT;  
END;  
/

**Scenario 2: Update Employee Bonus by Department**

***Stored Procedure: UpdateEmployeeBonus***

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (  
 dept\_id IN NUMBER,  
 bonus\_percent IN NUMBER  
) AS  
BEGIN  
 UPDATE Employees  
 SET Salary = Salary + (Salary \* (bonus\_percent / 100))  
 WHERE DepartmentID = dept\_id;  
 COMMIT;  
END;  
/

**Scenario 3: Transfer Funds Between Accounts**

***Stored Procedure: TransferFunds***

CREATE OR REPLACE PROCEDURE TransferFunds (  
 from\_acct IN NUMBER,  
 to\_acct IN NUMBER,  
 amount IN NUMBER  
) AS  
 from\_balance NUMBER;  
BEGIN  
 SELECT Balance INTO from\_balance FROM Accounts WHERE AccountID = from\_acct FOR UPDATE;  
  
 IF from\_balance >= amount THEN  
 UPDATE Accounts SET Balance = Balance - amount WHERE AccountID = from\_acct;  
 UPDATE Accounts SET Balance = Balance + amount WHERE AccountID = to\_acct;  
 COMMIT;  
 ELSE  
 RAISE\_APPLICATION\_ERROR(-20001, 'Insufficient balance in source account.');  
 END IF;  
END;  
/

Output:

A screenshot of a computer

AI-generated content may be incorrect.