PLSQL EXCERCISES:

Exercise 1: Control Structures

Scenario 1: Apply 1% Discount on Loan Interest for Customers Over 60

```
BEGIN
  FOR cust_rec IN (
    SELECT CustomerID, Age, LoanID, InterestRate
    FROM Customers
    JOIN Loans ON Customers.CustomerID = Loans.CustomerID
  ) LOOP
    IF cust_rec.Age > 60 THEN
      UPDATE Loans
      SET InterestRate = InterestRate - 1
      WHERE LoanID = cust_rec.LoanID;
    END IF;
  END LOOP;

  COMMIT;
END;
/
```

Scenario 2: Set IsVIP = TRUE for Customers with Balance > $10,000

```
BEGIN
  FOR cust_rec IN (SELECT CustomerID, Balance FROM Customers) LOOP
    IF cust_rec.Balance > 10000 THEN
      UPDATE Customers
      SET IsVIP = 'TRUE'
      WHERE CustomerID = cust_rec.CustomerID;
    END IF;
  END LOOP;
```

```
 COMMIT;
END;
/
```

Scenario 3: Send Reminders for Loans Due in Next 30 Days

```
DECLARE
 v_due_date DATE;
BEGIN
 FOR loan_rec IN (
  SELECT LoanID, CustomerID, DueDate
  FROM Loans
  WHERE DueDate BETWEEN SYSDATE AND SYSDATE + 30
 ) LOOP
  SELECT DueDate INTO v_due_date FROM Loans WHERE LoanID = loan_rec.LoanID;

  DBMS_OUTPUT.PUT_LINE('Reminder: Loan ID ' || loan_rec.LoanID ||
                ' for Customer ID ' || loan_rec.CustomerID ||
                ' is due on ' || TO_CHAR(v_due_date, 'DD-MON-YYYY'));
 END LOOP;
END;
/
```

Exercise 3: Stored Procedures

Scenario 1: ProcessMonthlyInterest Procedure

```
CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest AS
BEGIN
 FOR acc_rec IN (
  SELECT AccountID, Balance
  FROM Accounts
  WHERE AccountType = 'Savings'
 ) LOOP
```

```
    UPDATE Accounts
    SET Balance = Balance + (acc_rec.Balance * 0.01)
    WHERE AccountID = acc_rec.AccountID;
  END LOOP;

  COMMIT;
END;
/
```

Scenario 2: UpdateEmployeeBonus Procedure

```
CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(
  p_DepartmentID IN NUMBER,
  p_BonusPercent IN NUMBER  -- e.g., pass 10 for 10%
) AS
BEGIN
  UPDATE Employees
  SET Salary = Salary + (Salary * (p_BonusPercent / 100))
  WHERE DepartmentID = p_DepartmentID;

  COMMIT;
END;
/
```

Scenario 3: TransferFunds Procedure

```
CREATE OR REPLACE PROCEDURE TransferFunds(
  p_FromAccountID IN NUMBER,
  p_ToAccountID IN NUMBER,
  p_Amount IN NUMBER
) AS
  v_FromBalance NUMBER;
BEGIN
```

```
 -- Check balance of source account
 SELECT Balance INTO v_FromBalance
 FROM Accounts
 WHERE AccountID = p_FromAccountID;

 IF v_FromBalance < p_Amount THEN
   RAISE_APPLICATION_ERROR(-20001, 'Insufficient funds in source account.');
 END IF;

 -- Deduct from source
 UPDATE Accounts
 SET Balance = Balance - p_Amount
 WHERE AccountID = p_FromAccountID;

 -- Add to destination
 UPDATE Accounts
 SET Balance = Balance + p_Amount
 WHERE AccountID = p_ToAccountID;

 COMMIT;
END;
/
```