# Exercise 1: Control Structures - Integrated Questions and Realistic PL/SQL

## Initial Setup: Sample Tables and Data

Create Tables:  
```sql  
CREATE TABLE Customers (  
 CustomerID NUMBER PRIMARY KEY,  
 Name VARCHAR2(50),  
 Age NUMBER,  
 Balance NUMBER(10,2),  
 InterestRate NUMBER(5,2),  
 IsVIP VARCHAR2(5)  
);

CREATE TABLE Loans (  
 LoanID NUMBER PRIMARY KEY,  
 CustomerID NUMBER REFERENCES Customers(CustomerID),  
 DueDate DATE  
);  
```

Insert Sample Data:  
```sql  
INSERT INTO Customers VALUES (101, 'Alice', 65, 15000.00, 8.5, 'FALSE');  
INSERT INTO Customers VALUES (102, 'Bob', 45, 8500.00, 9.0, 'FALSE');  
INSERT INTO Customers VALUES (103, 'Carol', 70, 11000.00, 8.0, 'FALSE');  
INSERT INTO Customers VALUES (104, 'David', 30, 12000.00, 9.5, 'FALSE');

INSERT INTO Loans VALUES (201, 101, SYSDATE + 10);  
INSERT INTO Loans VALUES (202, 102, SYSDATE + 40);  
INSERT INTO Loans VALUES (203, 103, SYSDATE + 25);  
INSERT INTO Loans VALUES (204, 104, SYSDATE + 5);  
COMMIT;  
```

Alter Table (if needed):  
```sql  
ALTER TABLE Customers MODIFY IsVIP VARCHAR2(5);  
```

## Scenario 1: Discount for Senior Customers

Scenario:  
The bank wants to apply a discount to loan interest rates for customers above 60 years old.

Question:  
Write a PL/SQL block that loops through all customers, checks their age, and if they are above 60, apply a 1% discount to their current loan interest rates.

PL/SQL Code:  
```sql  
BEGIN  
 FOR cust\_rec IN (SELECT CustomerID, Age FROM Customers) LOOP  
 IF cust\_rec.Age > 60 THEN  
 UPDATE Customers  
 SET InterestRate = InterestRate \* 0.99  
 WHERE CustomerID = cust\_rec.CustomerID;  
 END IF;  
 END LOOP;  
 COMMIT;  
END;  
```

Output:  
- Alice and Carol are both above 60, so their interest rates will be reduced by 1%.

Verification Query:  
```sql  
SELECT Name, Age, InterestRate FROM Customers WHERE Age > 60;  
```

## Scenario 2: VIP Status Based on Balance

Scenario:  
A customer can be promoted to VIP status based on their balance.

Question:  
Write a PL/SQL block that iterates through all customers and sets a flag `IsVIP` to TRUE for those with a balance over $10,000.

PL/SQL Code:  
```sql  
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;  
```

Output:  
- Alice, Carol, and David have balances over $10,000, so their `IsVIP` field will be set to `TRUE`.

Verification Query:  
```sql  
SELECT Name, Balance, IsVIP FROM Customers WHERE IsVIP = 'TRUE';  
```

## Scenario 3: Loan Due Reminders

Scenario:  
The bank wants to send reminders to customers whose loans are due within the next 30 days.

Question:  
Write a PL/SQL block that fetches all loans due in the next 30 days and prints a reminder message for each customer.

PL/SQL Code:  
```sql  
DECLARE  
 v\_due\_date DATE;  
BEGIN  
 FOR loan\_rec IN (SELECT LoanID, CustomerID, DueDate  
 FROM Loans  
 WHERE DueDate BETWEEN SYSDATE AND SYSDATE + 30) LOOP  
 DBMS\_OUTPUT.PUT\_LINE('Reminder: Customer ' || loan\_rec.CustomerID ||  
 ', your loan ID ' || loan\_rec.LoanID ||  
 ' is due on ' || TO\_CHAR(loan\_rec.DueDate, 'DD-MON-YYYY'));  
 END LOOP;  
END;  
```

Output:  
Reminder: Customer 101, your loan ID 201 is due on [Date in 10 days]  
Reminder: Customer 103, your loan ID 203 is due on [Date in 25 days]  
Reminder: Customer 104, your loan ID 204 is due on [Date in 5 days]

Verification Query:  
```sql  
SELECT \* FROM Loans WHERE DueDate BETWEEN SYSDATE AND SYSDATE + 30;  
```