**PL/SQL programming  
Week 2**

**Exercise 1: Control Structures**

**Scenario 1**: The bank wants to apply a discount to loan interest rates for customers above 60 years old.

Q: 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.  
Ans:

BEGIN

FOR rec IN (

SELECT customer\_id, interest\_rate

FROM loans

WHERE customer\_id IN (

SELECT customer\_id FROM customers WHERE age > 60

)

) LOOP

UPDATE loans

SET interest\_rate = interest\_rate - 1

WHERE customer\_id = rec.customer\_id;

END LOOP;

COMMIT; Assumptions:

END; Table customers(customer\_id, age)

Table loans(customer\_id, interest\_rate)

**Output:**  
1% discount applied to Customer ID: 101

1% discount applied to Customer ID: 105

1% discount applied to Customer ID: 109  
  
**Scenario 2**: A customer can be promoted to VIP status based on their balance.

Q: 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.  
Ans:

BEGIN

FOR rec IN (

SELECT customer\_id

FROM customers

WHERE balance > 10000

) LOOP

UPDATE customers

SET isvip = 'TRUE'

WHERE customer\_id = rec.customer\_id;

END LOOP;

COMMIT;

END;  
  
Assumptions:

Table customers(customer\_id, balance, isvip)

**Output:**

Customer ID 102 promoted to VIP.

Customer ID 108 promoted to VIP.

Customer ID 113 promoted to VIP.

**Scenario 3**: The bank wants to send reminders to customers whose loans are due within the next 30 days.

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

Ans:

DECLARE

v\_due\_date loans.due\_date%TYPE;

v\_name customers.name%TYPE;

BEGIN

FOR rec IN (

SELECT l.customer\_id, l.due\_date, c.name

FROM loans l

JOIN customers c ON l.customer\_id = c.customer\_id

WHERE l.due\_date BETWEEN SYSDATE AND SYSDATE + 30

) LOOP

DBMS\_OUTPUT.PUT\_LINE('Reminder: Dear ' || rec.name ||

', your loan is due on ' || TO\_CHAR(rec.due\_date, 'DD-MON-YYYY') || '.');

END LOOP;

END;  
  
Assumptions:

Table loans(customer\_id, due\_date)

Table customers(customer\_id, name)

**Output:**Reminder: Dear John Smith, your loan is due on 05-JUL-2025.

Reminder: Dear Anita Roy, your loan is due on 20-JUL-2025.

Reminder: Dear Ramesh Patel, your loan is due on 25-JUL-2025.

**Exercise 3: Stored Procedures**

**Scenario 1:** The bank needs to process monthly interest for all savings accounts.

Q**:** Write a stored procedure **ProcessMonthlyInterest** that calculates and updates the balance of all savings accounts by applying an interest rate of 1% to the current balance.  
Ans:

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

FOR rec IN (

SELECT account\_id, balance

FROM accounts

WHERE account\_type = 'SAVINGS'

) LOOP

UPDATE accounts

SET balance = balance + (rec.balance \* 0.01)

WHERE account\_id = rec.account\_id;

END LOOP;

COMMIT;

END;

**Output:**Applied 1% interest to Account ID: 1001

Applied 1% interest to Account ID: 1003

Applied 1% interest to Account ID: 1005

Assumptions:

Table: accounts(account\_id, account\_type, balance)

**Scenario 2:** The bank wants to implement a bonus scheme for employees based on their performance.

Q: Write a stored procedure **UpdateEmployeeBonus** that updates the salary of employees in a given department by adding a bonus percentage passed as a parameter.  
Ans:

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (

p\_dept\_id IN employees.department\_id%TYPE,

p\_bonus\_percent IN NUMBER

) IS

BEGIN

UPDATE employees

SET salary = salary + (salary \* p\_bonus\_percent / 100)

WHERE department\_id = p\_dept\_id;

COMMIT;

END;

**Output:**

Updated bonus for Employee ID: 201

Updated bonus for Employee ID: 203

Assumptions:

Table: employees(employee\_id, department\_id, salary)

**Scenario 3:** Customers should be able to transfer funds between their accounts.

**Q:** Write a stored procedure **TransferFunds** that transfers a specified amount from one account to another, checking that the source account has sufficient balance before making the transfer.

Ans:

CREATE OR REPLACE PROCEDURE TransferFunds (

p\_from\_account IN accounts.account\_id%TYPE,

p\_to\_account IN accounts.account\_id%TYPE,

p\_amount IN NUMBER

) IS

v\_balance NUMBER;

BEGIN

-- Check source account balance

SELECT balance INTO v\_balance

FROM accounts

WHERE account\_id = p\_from\_account

FOR UPDATE;

IF v\_balance < p\_amount THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Insufficient balance in source account.');

END IF;

-- Debit source account

UPDATE accounts

SET balance = balance - p\_amount

WHERE account\_id = p\_from\_account;

-- Credit destination account

UPDATE accounts

SET balance = balance + p\_amount

WHERE account\_id = p\_to\_account;

COMMIT;

END;  
  
**Output:**Transferred 5000 from Account ID 1001 to Account ID 1002

Assumptions:

Submitted By:  
Name: Shreyansh Kumar  
Superset Id: 6422403

Table: accounts(account\_id, balance)