

WEEK 1 HANDS ON

R DARWIN RAJA KUMAR (312422205026)

St. Joseph's Institute of Technology

Exercise 1: Control Structures

Scenario 1: Discount on loan interest rates for customers above 60

```
BEGIN  FOR cust IN (SELECT CustomerID FROM Customers WHERE Age >
60) LOOP
    UPDATE Loans
    SET InterestRate = InterestRate - 1
WHERE CustomerID = cust.CustomerID;
END LOOP;
COMMIT;
END;
/
```

Scenario 2: Promote customers to VIP based on balance

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

Scenario 3: Send reminders for loans due in next 30 days

```
DECLARE
    CURSOR due_loans IS      SELECT
CustomerID, LoanID, DueDate
    FROM Loans      WHERE DueDate BETWEEN SYSDATE
AND SYSDATE + 30;

BEGIN
    FOR loan IN due_loans LOOP
        DBMS_OUTPUT.PUT_LINE('Reminder: Loan ID ' || loan.LoanID ||
                                ' for Customer ' || loan.CustomerID ||
' is due on ' || TO_CHAR(loan.DueDate, 'DD-MON-YYYY'));
    END LOOP;
END;
/
```

Exercise 2: Error Handling

Scenario 1: SafeTransferFunds procedure

```
CREATE OR REPLACE PROCEDURE SafeTransferFunds (  
  p_FromAcct IN NUMBER,  p_ToAcct  IN NUMBER,  
  p_Amount   IN NUMBER  
)  
IS  
  insufficient_funds EXCEPTION;  v_balance NUMBER; BEGIN  SELECT Balance INTO  
v_balance FROM Accounts WHERE AccountID = p_FromAcct FOR UPDATE;  
  
  IF v_balance < p_Amount THEN  
RAISE insufficient_funds;  
  END IF;  
  
  UPDATE Accounts SET Balance = Balance - p_Amount WHERE AccountID = p_FromAcct;  
UPDATE Accounts SET Balance = Balance + p_Amount WHERE AccountID = p_ToAcct;  
  
  COMMIT;  
  
EXCEPTION  WHEN  
insufficient_funds THEN  
  ROLLBACK;  DBMS_OUTPUT.PUT_LINE('Transfer failed:  
Insufficient funds.');
```

```
  WHEN OTHERS THEN  
  ROLLBACK;  DBMS_OUTPUT.PUT_LINE('Transfer failed:  
' || SQLERRM);  
END;  
/
```

Scenario 2: UpdateSalary procedure with error handling

```
CREATE OR REPLACE PROCEDURE UpdateSalary (  
  p_EmpID IN NUMBER,  p_Percent IN NUMBER )  
IS  
BEGIN  
  UPDATE Employees  SET Salary = Salary + (Salary  
* p_Percent / 100)  
  WHERE EmployeeID = p_EmpID;  
  
  IF SQL%ROWCOUNT = 0 THEN  RAISE_APPLICATION_ERROR(-20001,  
'Employee ID not found.');
```

```
  END IF;  
  
  COMMIT;  
  
EXCEPTION  
  WHEN OTHERS THEN  DBMS_OUTPUT.PUT_LINE('Error updating  
salary: ' || SQLERRM);  
END;  
/
```

Scenario 3: AddNewCustomer procedure with integrity check

```
CREATE OR REPLACE PROCEDURE AddNewCustomer (  
  p_CustomerID IN NUMBER,
```

```

    p_Name          IN VARCHAR2,
    p_Age           IN NUMBER,
    p_Balance       IN NUMBER )
IS
BEGIN
    INSERT INTO Customers (CustomerID, Name, Age, Balance)
VALUES (p_CustomerID, p_Name, p_Age, p_Balance);

    COMMIT;

EXCEPTION
    WHEN DUP_VAL_ON_INDEX THEN
        DBMS_OUTPUT.PUT_LINE('Insertion failed: Customer with ID ' || p_CustomerID || ' already
exists. ');    WHEN OTHERS THEN        DBMS_OUTPUT.PUT_LINE('Error inserting new customer: '
|| SQLERRM);
END;
/

```