SCHEMA CREATED:  
A screenshot of a computer

AI-generated content may be incorrect.

Exercise 1: Control Structures

Scenario 1:

**CODE:**

DECLARE

    v\_customer\_id Customers.CustomerID%TYPE;

    v\_dob DATE;

    v\_age NUMBER;

BEGIN

    FOR rec IN (SELECT CustomerID, DOB FROM Customers) LOOP

        v\_customer\_id := rec.CustomerID;

        v\_dob := rec.DOB;

        v\_age := TRUNC(MONTHS\_BETWEEN(SYSDATE, v\_dob) / 12);

        IF v\_age > 60 THEN

            UPDATE Loans

            SET InterestRate = InterestRate - 1

            WHERE CustomerID = v\_customer\_id;

            DBMS\_OUTPUT.PUT\_LINE('Interest rate discounted for Customer ID: ' || v\_customer\_id);

        END IF;

    END LOOP;

END;

/ **OUTPUT:**

**A screenshot of a computer

AI-generated content may be incorrect.**

Scenario 2:

**CODE:**

DECLARE

    v\_customer\_id Customers.CustomerID%TYPE;

    v\_balance Customers.Balance%TYPE;

BEGIN

    FOR rec IN (SELECT CustomerID, Balance FROM Customers) LOOP

        v\_customer\_id := rec.CustomerID;

        v\_balance := rec.Balance;

        IF v\_balance > 10000 THEN

            UPDATE Customers

            SET IsVIP = 'TRUE'

            WHERE CustomerID = v\_customer\_id;

            DBMS\_OUTPUT.PUT\_LINE('Customer ID ' || v\_customer\_id || ' promoted to VIP.');

        END IF;

    END LOOP;

END;

/

**OUTPUT:**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

Scenario 3:

**CODE:**

DECLARE

    v\_name Customers.Name%TYPE;

    v\_end\_date DATE;

BEGIN

    FOR rec IN (

        SELECT c.Name, l.EndDate

        FROM Loans l

        JOIN Customers c ON l.CustomerID = c.CustomerID

        WHERE l.EndDate BETWEEN SYSDATE AND SYSDATE + 30

    ) LOOP

        v\_name := rec.Name;

        v\_end\_date := rec.EndDate;

        DBMS\_OUTPUT.PUT\_LINE('Reminder: ' || v\_name || ', your loan is due on ' || TO\_CHAR(v\_end\_date, 'YYYY-MM-DD'));

    END LOOP;

END;

/

**OUTPUT:**

**A screenshot of a computer

AI-generated content may be incorrect.**

Exercise 3: Stored Procedures

Scenario 1:

**CODE:**

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

    FOR rec IN (

        SELECT AccountID, Balance

        FROM Accounts

        WHERE AccountType = 'Savings'

    ) LOOP

        UPDATE Accounts

        SET Balance = Balance + (Balance \* 0.01),

            LastModified = SYSDATE

        WHERE AccountID = rec.AccountID;

        DBMS\_OUTPUT.PUT\_LINE('Interest applied to Account ID: ' || rec.AccountID);

    END LOOP;

END;

/  
  
BEGIN

    ProcessMonthlyInterest;

END;

/

**OUTPUT:**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

Scenario 2:

**CODE:**

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(

p\_department IN Employees.Department%TYPE,

p\_bonus\_percent IN NUMBER

) IS

BEGIN

FOR rec IN (

SELECT EmployeeID, Salary

FROM Employees

WHERE Department = p\_department

) LOOP

UPDATE Employees

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

WHERE EmployeeID = rec.EmployeeID;

DBMS\_OUTPUT.PUT\_LINE('Bonus applied to Employee ID: ' || rec.EmployeeID);

END LOOP;

END;

/  
**BEGIN**

**UpdateEmployeeBonus('IT', 10); -- 10% bonus to IT department**

**END;**

**/**

**OUTPUT:**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

Scenario 3:

**CODE:**

CREATE OR REPLACE PROCEDURE TransferFunds(

p\_from\_account IN Accounts.AccountID%TYPE,

p\_to\_account IN Accounts.AccountID%TYPE,

p\_amount IN NUMBER

) IS

v\_balance NUMBER;

BEGIN

-- Check if the source account has sufficient balance

SELECT Balance INTO v\_balance

FROM Accounts

WHERE AccountID = p\_from\_account

FOR UPDATE;

IF v\_balance < p\_amount THEN

DBMS\_OUTPUT.PUT\_LINE('Insufficient balance in source account.');

RETURN;

END IF;

-- Deduct from source

UPDATE Accounts

SET Balance = Balance - p\_amount,

LastModified = SYSDATE

WHERE AccountID = p\_from\_account;

-- Add to destination

UPDATE Accounts

SET Balance = Balance + p\_amount,

LastModified = SYSDATE

WHERE AccountID = p\_to\_account;

DBMS\_OUTPUT.PUT\_LINE('Transferred ' || p\_amount ||

' from Account ' || p\_from\_account ||

' to Account ' || p\_to\_account);

END;

/

BEGIN

    TransferFunds(1, 2, 200);

END;

/

**OUTPUT:**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**