**Exercise 1: Control Structures**

**Scenario 1:**

BEGIN

  FOR cust IN (SELECT customer\_id, loan\_interest\_rate, age FROM customers) LOOP

    IF cust.age > 60 THEN

      UPDATE customers

      SET loan\_interest\_rate = loan\_interest\_rate - 1

      WHERE customer\_id = cust.customer\_id;

    END IF;

  END LOOP;

  COMMIT;

END;

**Output:**

A computer screen shot of a computer

AI-generated content may be incorrect.

SELECT \* FROM customers;

A screenshot of a computer

AI-generated content may be incorrect.

**Scenario 2:**

BEGIN

  FOR cust IN (SELECT customer\_id, balance FROM customers) LOOP

    IF cust.balance > 10000 THEN

      UPDATE customers

      SET isvip = 'TRUE'

      WHERE customer\_id = cust.customer\_id;

    END IF;

  END LOOP;

  COMMIT;

END;

**Output:**

A computer screen shot of a computer screen

AI-generated content may be incorrect.

SELECT customer\_id, name, isvip FROM customers;

A screenshot of a computer

AI-generated content may be incorrect.

**Scenario 3:**

SET SERVEROUTPUT ON;

BEGIN

  FOR loan IN (

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

    FROM loans l

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

    WHERE l.due\_date <= SYSDATE + 30

  ) LOOP

    DBMS\_OUTPUT.PUT\_LINE('Reminder: Loan ' || loan.loan\_id || ' for customer ' || loan.name ||

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

  END LOOP;

END;

**Output:**

A black text on a white background

AI-generated content may be incorrect.

**Exercise 3: Stored Procedures**

**Scenario 1:**

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest AS

BEGIN

  FOR acc IN (SELECT account\_id, balance FROM savings\_accounts) LOOP

    UPDATE savings\_accounts

    SET balance = balance + (balance \* 0.01)

    WHERE account\_id = acc.account\_id;

  END LOOP;

  COMMIT;

END;

**A computer screen shot of a computer code

AI-generated content may be incorrect.**

**A screenshot of a computer screen

AI-generated content may be incorrect.**

**Scenario 2:**

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (

  dept\_id IN NUMBER,

  bonus\_percent IN NUMBER

) AS

BEGIN

  UPDATE employees

  SET salary = salary + (salary \* bonus\_percent / 100)

  WHERE department\_id = dept\_id;

  COMMIT;

END;

**A white background with text

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Scenario 3:**

CREATE OR REPLACE PROCEDURE TransferFunds (

  from\_account IN NUMBER,

  to\_account IN NUMBER,

  amount IN NUMBER

) AS

  insufficient\_balance EXCEPTION;

BEGIN

  -- Check balance

  DECLARE

    v\_balance NUMBER;

  BEGIN

    SELECT balance INTO v\_balance FROM accounts WHERE account\_id = from\_account;

    IF v\_balance < amount THEN

      RAISE insufficient\_balance;

    END IF;

    -- Deduct from source

    UPDATE accounts

    SET balance = balance - amount

    WHERE account\_id = from\_account;

    -- Add to target

    UPDATE accounts

    SET balance = balance + amount

    WHERE account\_id = to\_account;

    COMMIT;

  END;

EXCEPTION

  WHEN insufficient\_balance THEN

    DBMS\_OUTPUT.PUT\_LINE('Transfer failed: Insufficient funds.');

  WHEN OTHERS THEN

    DBMS\_OUTPUT.PUT\_LINE('Transfer failed: ' || SQLERRM);

END;

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer code

AI-generated content may be incorrect.**