PL/SQL Exercises — Questions with Solutions

Question 1

A company wants to calculate the annual salary of an employee. Write a PL/SQL block that

```
SET SERVEROUTPUT ON;
DECLARE
  v_basic_salary NUMBER := 2000;
  v_bonus        NUMBER := 1500;
  v_annual_salary NUMBER;
BEGIN
  v_annual_salary := v_basic_salary * 12 + v_bonus;
  DBMS_OUTPUT.PUT_LINE('Annual Salary: ' || v_annual_salary);
END;
/
```

Question 2

A university stores a student's marks in 3 subjects. Write a PL/SQL block to calculate the average marks and display the result.

```
SET SERVEROUTPUT ON;
DECLARE
  m1 NUMBER := 78;
  m2 NUMBER := 85;
  m3 NUMBER := 90;
  avg_marks NUMBER;
BEGIN
  avg_marks := (m1 + m2 + m3) / 3;
  DBMS_OUTPUT.PUT_LINE('Average Marks: ' || TO_CHAR(avg_marks));
END;
/
```

Question 3

A bank system stores a customer's account balance.
If balance < 1000 → print "Low Balance"
If balance between 1000 and 5000 → print "Sufficient Balance"
If balance > 5000 → print "High Balance"
Write a PL/SQL block using IF-ELSIF.

```
SET SERVEROUTPUT ON;
DECLARE
  balance NUMBER := 2750; -- example
BEGIN
```

```
  IF balance < 1000 THEN
    DBMS_OUTPUT.PUT_LINE('Low Balance');
  ELSIF balance BETWEEN 1000 AND 5000 THEN
    DBMS_OUTPUT.PUT_LINE('Sufficient Balance');
  ELSE
    DBMS_OUTPUT.PUT_LINE('High Balance');
  END IF;
END;
/
```

Question 4

A grading system accepts a student's percentage.
90–100 → "A Grade"
75–89 → "B Grade"
50–74 → "C Grade"
Below 50 → "Fail"
Write using a CASE statement.

```
SET SERVEROUTPUT ON;
DECLARE
  pct NUMBER := 83; -- example percentage
  result VARCHAR2(20);
BEGIN
  CASE
    WHEN pct BETWEEN 90 AND 100 THEN
      result := 'A Grade';
    WHEN pct BETWEEN 75 AND 89 THEN
      result := 'B Grade';
    WHEN pct BETWEEN 50 AND 74 THEN
      result := 'C Grade';
    ELSE
      result := 'Fail';
  END CASE;
  DBMS_OUTPUT.PUT_LINE('Result: ' || result);
END;
/
```

Question 5

A shopping store gives discounts:
If the bill > 5000 → 20% discount
If the bill between 2000 and 5000 → 10% discount
Otherwise no discount
Write a PL/SQL block to calculate final bill after discount.

```
SET SERVEROUTPUT ON;
DECLARE
```

```
  bill_amount NUMBER := 4200; -- example
  discount_rate NUMBER := 0;
  final_bill NUMBER;
BEGIN
  IF bill_amount > 5000 THEN
    discount_rate := 0.20;
  ELSIF bill_amount BETWEEN 2000 AND 5000 THEN
    discount_rate := 0.10;
  ELSE
    discount_rate := 0;
  END IF;

  final_bill := bill_amount * (1 - discount_rate);
  DBMS_OUTPUT.PUT_LINE('Original Bill: ' || bill_amount);
  DBMS_OUTPUT.PUT_LINE('Discount Rate: ' || (discount_rate*100) ||
'%');
  DBMS_OUTPUT.PUT_LINE('Final Bill: ' || final_bill);
END;
/
```

Question 6

Write a PL/SQL block that prints the multiplication table of a number entered by the user (example: table of 7).

```
SET SERVEROUTPUT ON;
DECLARE
  num NUMBER := 7; -- change as needed
BEGIN
  FOR i IN 1..10 LOOP
    DBMS_OUTPUT.PUT_LINE(num || ' x ' || i || ' = ' || (num * i));
  END LOOP;
END;
/
```

Question 7

A company wants to print employee IDs from 100 to 120. Use a FOR LOOP to print them.

```
SET SERVEROUTPUT ON;
BEGIN
  FOR emp_id IN 100..120 LOOP
    DBMS_OUTPUT.PUT_LINE('Employee ID: ' || emp_id);
  END LOOP;
END;
/
```

Question 8

Write a PL/SQL block to display the factorial of a given number using a WHILE loop.

```
SET SERVEROUTPUT ON;
DECLARE
  n NUMBER := 6; -- example
  i NUMBER := 1;
  fact NUMBER := 1;
BEGIN
  WHILE i <= n LOOP
    fact := fact * i;
    i := i + 1;
  END LOOP;
  DBMS_OUTPUT.PUT_LINE('Factorial of ' || n || ' is ' || fact);
END;
/
```

Question 9

A countdown timer should print numbers from 10 down to 1 using a REVERSE FOR loop.

```
SET SERVEROUTPUT ON;
BEGIN
  FOR i IN REVERSE 1..10 LOOP
    DBMS_OUTPUT.PUT_LINE(i);
  END LOOP;
END;
/
```

Question 10

Print the names of all employees in the IT department using a FOR loop with a SELECT query. (Assume departments table exists with dept_name)

```
SET SERVEROUTPUT ON;
BEGIN
  FOR rec IN (
    SELECT e.emp_name
    FROM employees e
    JOIN departments d ON e.dept_id = d.dept_id
    WHERE d.dept_name = 'IT'
  ) LOOP
    DBMS_OUTPUT.PUT_LINE('Employee: ' || rec.emp_name);
  END LOOP;
END;
/
```

Question 11

Give a 10% salary increase to all employees whose salary < 3000. Use a loop to update salaries.

```
SET SERVEROUTPUT ON;
DECLARE
BEGIN
  FOR rec IN (SELECT emp_id, salary FROM employees WHERE salary < 3000)
LOOP
    UPDATE employees
    SET salary = ROUND(rec.salary * 1.10, 2)
    WHERE emp_id = rec.emp_id;
    DBMS_OUTPUT.PUT_LINE('Updated emp_id ' || rec.emp_id || ' to ' ||
ROUND(rec.salary * 1.10,2));
  END LOOP;
  COMMIT;
END;
/
```

Question 12

Display all employees whose salary is above the average salary of the company.

```
SET SERVEROUTPUT ON;
DECLARE
  avg_sal NUMBER;
BEGIN
  SELECT AVG(salary) INTO avg_sal FROM employees;

  DBMS_OUTPUT.PUT_LINE('Average Salary: ' || TO_CHAR(avg_sal));
  FOR rec IN (SELECT emp_id, emp_name, salary FROM employees WHERE
salary > avg_sal) LOOP
    DBMS_OUTPUT.PUT_LINE(rec.emp_id || ' - ' || rec.emp_name || ' : '
|| rec.salary);
  END LOOP;
END;
/
```

Question 13

Write a PL/SQL block that prints:
'High Earner' if salary > 8000
'Mid Earner' if salary between 4000–8000
'Low Earner' otherwise.

```
SET SERVEROUTPUT ON;
BEGIN
  FOR rec IN (SELECT emp_id, emp_name, salary FROM employees) LOOP
```

```
    IF rec.salary > 8000 THEN
      DBMS_OUTPUT.PUT_LINE(rec.emp_name || ' (' || rec.emp_id || '):
High Earner');
    ELSIF rec.salary BETWEEN 4000 AND 8000 THEN
      DBMS_OUTPUT.PUT_LINE(rec.emp_name || ' (' || rec.emp_id || '):
Mid Earner');
    ELSE
      DBMS_OUTPUT.PUT_LINE(rec.emp_name || ' (' || rec.emp_id || '):
Low Earner');
    END IF;
  END LOOP;
END;
/
```

Question 14

Write a PL/SQL program that prints the total salary cost of each department (group by dept_id).

```
SET SERVEROUTPUT ON;
BEGIN
  FOR rec IN (
    SELECT dept_id, SUM(salary) AS total_salary
    FROM employees
    GROUP BY dept_id
  ) LOOP
    DBMS_OUTPUT.PUT_LINE('Dept ' || rec.dept_id || ' - Total Salary: '
|| rec.total_salary);
  END LOOP;
END;
/
```

Question 15

Write a PL/SQL block that accepts a number n and prints the Fibonacci sequence up to n terms.

```
SET SERVEROUTPUT ON;
DECLARE
  n NUMBER := 10; -- number of terms
  a NUMBER := 0;
  b NUMBER := 1;
  temp NUMBER;
  i NUMBER := 1;
BEGIN
  IF n <= 0 THEN
    DBMS_OUTPUT.PUT_LINE('Please enter a positive integer');
  ELSIF n = 1 THEN
    DBMS_OUTPUT.PUT_LINE(a);
```

```
  ELSE
    DBMS_OUTPUT.PUT_LINE('Fibonacci sequence up to ' || n || '
terms:');
    DBMS_OUTPUT.PUT_LINE(a);
    DBMS_OUTPUT.PUT_LINE(b);
    i := 3;
    WHILE i <= n LOOP
      temp := a + b;
      DBMS_OUTPUT.PUT_LINE(temp);
      a := b;
      b := temp;
      i := i + 1;
    END LOOP;
  END IF;
END;
/
```

Question 16

A bank wants to process 100 transactions stored in a table transactions(txn_id, amount, type) where type = 'CREDIT' or 'DEBIT'. Write a PL/SQL block that calculates final account balance after all transactions.

```
SET SERVEROUTPUT ON;
DECLARE
  v_balance NUMBER := 0; -- starting balance (change if needed)
BEGIN
  FOR rec IN (SELECT txn_id, amount, type FROM transactions ORDER BY
txn_id) LOOP
    IF UPPER(rec.type) = 'CREDIT' THEN
      v_balance := v_balance + rec.amount;
    ELSIF UPPER(rec.type) = 'DEBIT' THEN
      v_balance := v_balance - rec.amount;
    END IF;
  END LOOP;
  DBMS_OUTPUT.PUT_LINE('Final Account Balance: ' || v_balance);
END;
/
```

Question 17

Write a PL/SQL procedure that takes an employee ID and prints:
Employee Name
Department Name
Current Salary

```
CREATE OR REPLACE PROCEDURE print_employee_info(p_emp_id IN
employees.emp_id%TYPE) IS
```

```
  v_name employees.emp_name%TYPE;
  v_dept departments.dept_name%TYPE;
  v_salary employees.salary%TYPE;
BEGIN
  SELECT e.emp_name, d.dept_name, e.salary
    INTO v_name, v_dept, v_salary
    FROM employees e
    LEFT JOIN departments d ON e.dept_id = d.dept_id
   WHERE e.emp_id = p_emp_id;

  DBMS_OUTPUT.PUT_LINE('Employee Name: ' || v_name);
  DBMS_OUTPUT.PUT_LINE('Department: ' || NVL(v_dept, 'N/A'));
  DBMS_OUTPUT.PUT_LINE('Current Salary: ' || v_salary);
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('No employee found with ID ' || p_emp_id);
END print_employee_info;
/
```