Name: Mohammad Omid Musleh Father Name: Fazl Khuda

1. Basic & Variables

1. A company wants to calculate the annual salary of an employee. Write a PL/SQL block that takes basic_salary and bonus as variables and prints the annual salary.

```
DECLARE
        basic_salary NUMBER := 5000;
        bonus NUMBER := 1000;
        annual_salary NUMBER;
BEGIN
        annual_salary := (basic_salary  + bonus) * 12;
        DBMS_OUTPUT.PUT_LINE('Annual Salary: '|| annual_salary);
END;
/
```

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.

```
DECLARE
        mark1 NUMBER := 85;
        mark2 NUMBER := 90;
        mark3 NUMBER := 78;
        average NUMBER;
BEGIN
        average := (mark1 + mark2 +mark3) / 3;
        DBMS_OUTPUT.PUT_LINE('Average Marks: '|| ROUND(average, 2));
END;
```

2. Conditional Statements

3. A bank system stores a customer's accounts 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

```
DECLARE
        balance NUMBER := 3000;
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;
/
```

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 statements.

```
DECLARE
        percentage NUMBER := 80;
        grade VARCHAR2(10);
BEGIN
        grade := CASE
                WHEN percentage BETWEEN 90 AND 100 THEN 'A Grade'
                WHEN percentage BETWEEN 75 AND 89 THEN 'B Grade'
                WHEN percentage BETWEEN 50 AND 74 THEN 'C Grade'
                ELSE 'Fail'
        END;
        DBMS_OUTPUT.PUT_LINE('Grade: '|| grade);
END;
/
```

5. A shopping store gives discount:

- 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.

```
DECLARE
        bill_amount NUMBER := 6000;
        discount NUMBER;
        final_bill NUMBER;
BEGIN
        IF bill_amount > 5000 THEN
                Discount := 0.20;
        ELSIF bill_amount BETWEEN 2000 AND 5000 THEN
                Discount := 0.10;
        ELSE
                Discount := 0;
        END IF;
        Final_bill := bill_amount – (bill_amount * discount);
        DBMS_OUTPUT.PUT_LINE('Final Bill: '|| final_bill);
END;
/
```

3. Looping

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

```
DECLARE
        n NUMBER := 7;
        i NUMBER;
BEGIN
        FOR i IN 1..10 LOOP
                DBMS_OUTPUT.PUT_LINE(n || ‘ x ’ || i || ‘ = ’ || n*i);
        END LOOP;
END;
/
```

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

```
BEGIN
        FOR emp_id IN 100..120 LOOP
                DBMS_OUTPUT.PUT_LINE(‘Employee ID: ’ || emp_id);
        END LOOP;
END;
/
```

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

```
DECLARE
        n NUMBER := 5;
        factorial NUMBER := 1;
        i NUMBER := 1;
BEGIN
        WHILE I <= n LOOP
                factorial := factorial * i;
                i := i + 1;
        END LOOP;
        DBMS_OUTPUT.PUT_LINE('Factorial of ' || n || ' is ' || factorial);
END;
/
```

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

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

4. Table-Based Scenario (using employee table)

(Assume table employees(emp_id, emp_name, salary, dept_id) exists)

10. Print the names of all employee in the IT department using a FOR loop with a SELECT query.

```
DECLARE
    CURSOR c_emp IS
        SELECT emp_name FROM employees WHERE dept_id = 'IT';
BEGIN
    FOR emp_rec IN c_emp LOOP
        DBMS_OUTPUT.PUT_LINE(emp_rec.emp_name);
    END LOOP;
END;
/
```

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

```
BEGIN
    FOR emp_rec IN (SELECT emp_id, salary FROM employees WHERE
    salary < 3000) LOOP
        UPDATE employees
        SET salary = salary * 1.10
        WHERE emp_id = emp_rec.emp_id;
    END LOOP;
    COMMIT;
END;
/
```

12. Display all employee whose salary is above the average salary of the company.

```
DECLARE
        avg_salary NUMBER;
BEGIN
        SELECT AVG(salary) INTO avg_salary FROM employees;
        FOR emp_rec IN (SELECT emp_name, salary FROM employees
        WHERE salary > avg_salary);
                DBMS_OUTPUT.PUT_LINE(emp_rec.emp_name || ‘ - ’ ||
emp_rec.salary);
        END LOOP;
END;
/
```

13. Write a PL/SQL block that prints:

- “High Earner” if salary > 8000
- “Mid Earner” if salary between 4000-8000
- “Low Earner” otherwise.

```
DECLARE
        CURSOR c_emp IS SELECT emp_name, salary FROM employees;
BEGIN
        FOR emp_rec IN c_emp LOOP
                IF emp_rec.salary > 8000 THEN
                        DBMS_OUTPUT.PUT_LINE(emp_rec.emp_name || ‘ – High Earner’);
                ELSIF emp_rec.salary BERWEEN 4000 AND 8000 THEN
                        DBMS_OUTPUT.PUT_LINE(emp_rec.emp_name || ‘ – Mid Earner’);
                ELSE
                        DBMS_OUTPUT.PUT_LINE(emp_rec.emp_name || ‘ – Low Earner’);
                END IF;
        END LOOP;
END;
/
```

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

```
DECLARE
    CURSOR c_dept IS
        SELECT dept_id, SUM(salary) AS total_salary
        FROM employees
        GROUP BY dept_id;
BEGIN
    FOR emp_rec IN c_dept LOOP
        DBMS_OUTPUT.PUT_LINE('Department: ' || || dept_rec.dept_id
|| ', Total Salary: ' || dept_rec.total_salary);
    END LOOP;
END;
/
```

5. Challenge Level🚀

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

```
DECLARE
        n NUMBER := 10;
        a NUMBER := 0;
        b NUMBER := 1;
        temp NUMBER;
        i NUMBER;
BEGIN
        DBMS_OUTPUT.PUT_LINE('Fibonacci Sequence: ');
        DBMS_OUTPUT.PUT_LINE(a);
        DBMS_OUTPUT.PUT_LINE(b);

        FOR i IN 3..n LOOP
                temp := a + b;
                a := b;
                b:= temp;
                DBMS_OUTPUT.PUT_LINE(temp);
        END LOOP;
END;
/
```

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.

```
DECLARE
    Final_balance NUMBER := 0;
BEGIN
    FOR txn_rec IN (SELECT amount, type FROM transactions ORDER BY txn_id) LOOP
        IF txn_rec.type = ‘CREDIT’ THEN
            final_balance := final_balance + txn_rec.amount;
        ELSIF txn_rec.type = ‘DEBIT’ THEN
            final_balance := final_balance - txn_rec.amount;
        END IF;
    END LOOP;
    DBMS_OUTPUT.PUT_LINE(‘Final Balance: ’ || final_balance);
END;
/
```

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

```
CREATE OR REPLACE PROCEDURE get_employee_info(p_emp_id IN
employees.emp_id%TYPE) IS
        v_emp_name employees.emp_name%TYPE;
        v_dept_name departments.dept_name%TYPE;
        v_salary employees.salary%TYPE;
BEGIN
        SELECT e.emp_name, d.dept_name, e.salary
        INTO v_temp_name, v_dept_name, v_salary
        FROM employees e
        JOIN departments d ON e.dept_id = d.dept_id
        WHERE e.emp_id = p_emp_id;

        DBMS_OUTPUT.PUT_LINE('Employee Name: ' || v_emp_name);
        DBMS_OUTPUT.PUT_LINE('Department Name: ' || v_dept_name);
        DBMS_OUTPUT.PUT_LINE('Current Salary: ' || v_salary);
EXCEPTION
        WHEN NO_DATA_FOUND THEN
                DBMS_OUTPUT.PUT_LINE('Employee not found! ');
END;
/
-----------------------------------------------------------------------------------------------
BEGIN
        get_employee_info(100);
END;
/
```