PL/SQL Exercises

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.

Answer 1:

DECLARE

basic\_salary NUMBER := 50000;

bonus NUMBER := 10000;

annual\_salary NUMBER;

BEGIN

annual\_salary:= (basic\_salary \* 12) + bonus;

DBMS\_OUTPUT.PUT\_LINE('Annual Salary is: ' || 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.

Answer2:

DECLARE

mark1 NUMBER := 85;

mark2 NUMBER := 90;

mark3 NUMBER := 88;

average\_marks NUMBER;

BEGIN

average\_marks := (mark1 + mark2 + mark3) / 3;

DBMS\_OUTPUT.PUT\_LINE('Average Marks: ' || TO\_CHAR(ROUND(average\_marks, 2)));

END;

/

2. Conditional Statements

3. A bank system stores a customer’s account balance.

o If balance < 1000 → print "Low Balance"

o If balance between 1000 and 5000 → print "Sufficient Balance"

o If balance > 5000 → print "High Balance"

Write a PL/SQL block using IF-ELSIF.

Answer3 :

DECLARE

balance NUMBER := 3200;

BEGIN

IF balance < 1000 THEN

DBMS\_OUTPUT.PUT\_LINE('Low Balance');

ELSIF balance >= 1000 AND balance <= 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.

o 90–100 → "A Grade"

o 75–89 → "B Grade"

o 50–74 → "C Grade"

o Below 50 → "Fail"

Write using a CASE statement.

Answer5:

DECLARE

percentage NUMBER := 82;

grade VARCHAR2(20);

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'

WHEN percentage < 50 THEN 'Fail'

ELSE 'Invalid Percentage'

END;

DBMS\_OUTPUT.PUT\_LINE('Percentage: ' || percentage || ' → ' || grade);

END;

/

5. A shopping store gives discounts:

o If the bill > 5000 → 20% discount

o If the bill between 2000 and 5000 → 10% discount

o Otherwise no discount

Write a PL/SQL block to calculate final bill after discount.

Answer5:

DECLARE

bill\_amount NUMBER := 4500;

discount NUMBER := 0;

final\_amount NUMBER;

BEGIN

IF bill\_amount > 5000 THEN

discount := bill\_amount \* 0.20;

ELSIF bill\_amount BETWEEN 2000 AND 5000 THEN

discount := bill\_amount \* 0.10;

ELSE

discount := 0;

END IF;

final\_amount := bill\_amount - discount;

DBMS\_OUTPUT.PUT\_LINE('Original Bill: ' || bill\_amount);

DBMS\_OUTPUT.PUT\_LINE('Discount: ' || discount);

DBMS\_OUTPUT.PUT\_LINE('Final Amount to Pay: ' || final\_amount);

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).

Answer6:

DECLARE

num NUMBER := 7;

i NUMBER := 1;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Multiplication Table of ' || num);

WHILE i <= 10 LOOP

DBMS\_OUTPUT.PUT\_LINE(num || ' x ' || i || ' = ' || (num \* i));

i := i + 1;

END LOOP;

END;

/

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

Answer7:

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.

Answer8:

DECLARE

num NUMBER := 6;

factorial NUMBER := 1;

i NUMBER := 1;

BEGIN

WHILE i <= num LOOP

factorial := factorial \* i;

i := i + 1;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('Factorial of ' || num || ' is ' || factorial);

END;

/

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

Answer9:

BEGIN

FOR i IN REVERSE 1..10 LOOP

DBMS\_OUTPUT.PUT\_LINE(i);

END LOOP;

END;

/

4. Table-Based Scenarios (using employees table)

(Assume table employees(emp\_id, emp\_name, salary, dept\_id) exists)

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

Answer10:

BEGIN

FOR emp\_record IN (

SELECT emp\_name

FROM employees

WHERE dept\_id = 10

) LOOP

DBMS\_OUTPUT.PUT\_LINE(emp\_record.emp\_name);

END LOOP;

END;

/

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

Answer11:

DECLARE

CURSOR low\_salary\_emp IS

SELECT emp\_id, salary

FROM employees

WHERE salary < 3000

FOR UPDATE;

BEGIN

FOR emp\_rec IN low\_salary\_emp LOOP

UPDATE employees

SET salary = salary \* 1.10

WHERE CURRENT OF low\_salary\_emp;

END LOOP;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Salaries updated for employees earning less than 3000.');

END;

/

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

Answer12:

DECLARE

avg\_salary NUMBER;

BEGIN

SELECT AVG(salary) INTO avg\_salary FROM employees;

DBMS\_OUTPUT.PUT\_LINE('Average Salary: ' || ROUND(avg\_salary, 2));

DBMS\_OUTPUT.PUT\_LINE('Employees with salary above average:');

-- Loop through employees with salary above average

FOR emp\_rec IN (

SELECT emp\_name, salary

FROM employees

WHERE salary > avg\_salary

) LOOP

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.

Answer13:

DECLARE

salary NUMBER := 6500;

category VARCHAR2(20);

BEGIN

IF salary > 8000 THEN

category := 'High Earner';

ELSIF salary BETWEEN 4000 AND 8000 THEN

category := 'Mid Earner';

ELSE

category := 'Low Earner';

END IF;

DBMS\_OUTPUT.PUT\_LINE('Salary: ' || salary || ' → ' || category);

END;

/

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

Answer14:

BEGIN

FOR dept\_rec IN (

SELECT dept\_id, SUM(salary) AS total\_salary

FROM employees

GROUP BY dept\_id

ORDER BY dept\_id

) LOOP

DBMS\_OUTPUT.PUT\_LINE('Department ID: ' || dept\_rec.dept\_id || ' - Total Salary Cost: ' || dept\_rec.total\_salary);

END LOOP;

END;

/

5. Challenge Level

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

Answer15:

DECLARE

n NUMBER := 10;

a NUMBER := 0;

b NUMBER := 1;

temp NUMBER;

count NUMBER := 1;

BEGIN

IF n <= 0 THEN

DBMS\_OUTPUT.PUT\_LINE('Please enter a positive integer.');

ELSIF n = 1 THEN

DBMS\_OUTPUT.PUT\_LINE('Fibonacci sequence up to 1 term:');

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);

WHILE count < n - 2 LOOP

temp := a + b;

DBMS\_OUTPUT.PUT\_LINE(temp);

a := b;

b := temp;

count := count + 1;

END LOOP;

END IF;

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.

Answer16:

DECLARE

final\_balance NUMBER := 0;

BEGIN

FOR txn\_rec IN (

SELECT txn\_id, amount, type

FROM transactions

WHERE ROWNUM <= 100

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 Account Balance after 100 transactions: ' || final\_balance);

END;

/

17. Write a PL/SQL procedure that takes an employee ID and prints:

Employee Name

Department Name

Current Salary

Answer17:

CREATE OR REPLACE PROCEDURE print\_employee\_details(p\_emp\_id IN NUMBER) IS

v\_emp\_name employees.emp\_name%TYPE;

v\_salary employees.salary%TYPE;

v\_dept\_name VARCHAR2(100);

BEGIN

SELECT e.emp\_name, e.salary, d.dept\_name

INTO v\_emp\_name, v\_salary, v\_dept\_name

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('No employee found with ID ' || p\_emp\_id);

END;

/