SQL ASSIGNMENT NO. 4 .

1) Write a PL/SQL code to display total marks of six subject, percentage gets with a division.

If percentage greater than or equal to 60 get First division

If percentage greater than or equal to 50 or less than 60 get Second division

If percentage greater than or equal to 40 or less than 50 get Third division

If percentage less than 40 Fail

-->


```
SQL> DECLARE
  2     s1 INT := &s1;
  3     s2 INT := &s2;
  4     s3 INT := &s3;
  5     s4 INT := &s4;
  6     s5 INT := &s5;
  7     s6 INT := &s6;
  8
  9     total  INT;
 10     per    INT;
 11     div    VARCHAR2(20);
 12  BEGIN
 13     total := s1 + s2 + s3 + s4 + s5 + s6;
 14     per := (total / 600) * 100;
```

```
15
16      div := CASE
17          WHEN per >= 60 THEN 'First Division'
18          WHEN per >= 50 THEN 'Second Division'
19          WHEN per >= 40 THEN 'Third Division'
20          ELSE 'Fail'
21      END;
22
23      DBMS_OUTPUT.PUT_LINE('Total Marks: ' || total);
24      DBMS_OUTPUT.PUT_LINE('Percentage: ' || per || '%');
25      DBMS_OUTPUT.PUT_LINE('Division: ' || div);
26  END;
27  /
```

```
Enter value for s1: 60
Enter value for s2: 70
Enter value for s3: 75
Enter value for s4: 66
Enter value for s5: 85
Enter value for s6: 90
Total Marks: 446
Percentage: 74%
Division: First Division
```

2) Write a PL/SQL code for calculate the cost of the apple box. If the weight of apple box

is greater than equal to 15 kg, rate is Rs. 7/kg and weight is less than 10 kg, rate is Rs. 9/kg.

-->

```
SQL> DECLARE
  2     wt   INT := &wt;
  3     r    INT;
  4     c    INT;
  5  BEGIN
  6     IF wt >= 15 THEN
  7        r := 7;
  8     ELSIF wt < 10 THEN
  9        r := 9;
 10     ELSE
 11        r := 8;
 12     END IF;
 13
 14     c := wt * r;
 15
 16     DBMS_OUTPUT.PUT_LINE('Weight of Apple Box: ' || wt || ' kg');
 17     DBMS_OUTPUT.PUT_LINE('Rate: Rs. ' || r || '/kg');
 18     DBMS_OUTPUT.PUT_LINE('Total Cost: Rs. ' || c);
 19  END;
 20  /
```

Enter value for wt: 10

Weight of Apple Box: 10 kg

Rate: Rs. 8/kg

Total Cost: Rs. 80

PL/SQL procedure successfully completed.

SQL> /

Enter value for wt: 16

Weight of Apple Box: 16 kg

Rate: Rs. 7/kg

Total Cost: Rs. 112

PL/SQL procedure successfully completed.

3) Write a PL/SQL code to calculate the net salary of an employee in a particular month

considering various allowances (TA, DA, HRA) and deductions (INCOME TAX,

PROVIDEND FUND) as:

TA=15 percent of basic salary, DA=2 percent of basic salary

HRA=10 percent of basic salary, INCOME TAX=5 percent of salary

PROVIDEND FUND=10 percent of salary

-->

```
SQL> DECLARE
  2     bs NUMBER := &bs;
  3     ta            NUMBER;
  4     da            NUMBER;
  5     hra           NUMBER;
  6     income_tax    NUMBER;
  7     pro_fund      NUMBER;
  8     total_salary  NUMBER;
  9     net_salary    NUMBER;
 10  BEGIN
 11     ta := bs * 0.15;
 12     da := bs * 0.02;
 13     hra := bs * 0.10;
 14     total_salary := bs + ta + da + hra;
 15
 16     income_tax := total_salary * 0.05;
 17     pro_fund := total_salary * 0.10;
 18
 19     net_salary := total_salary - (income_tax + pro_fund);
 20
 21     DBMS_OUTPUT.PUT_LINE('Basic Salary: Rs. ' || bs);
 22     DBMS_OUTPUT.PUT_LINE('Travel Allowance: Rs. ' || ta);
 23     DBMS_OUTPUT.PUT_LINE('Dearness Allowance: Rs. ' || da);
 24     DBMS_OUTPUT.PUT_LINE('House Rent Allowance ): Rs. ' || hra);
 25     DBMS_OUTPUT.PUT_LINE('Total Salary: Rs. ' || total_salary);
```

```
26     DBMS_OUTPUT.PUT_LINE('Income Tax: Rs. ' || income_tax);
27     DBMS_OUTPUT.PUT_LINE('Provident Fund: Rs. ' || pro_fund);
28     DBMS_OUTPUT.PUT_LINE('Net Salary: Rs. ' || net_salary);
29  END;
30  /
Enter value for bs: 5000
Basic Salary: Rs. 5000
Travel Allowance: Rs. 750
Dearness Allowance: Rs. 100
House Rent Allowance ): Rs. 500
Total Salary: Rs. 6350
Income Tax: Rs. 317.5
Provident Fund: Rs. 635
Net Salary: Rs. 5397.5

PL/SQL procedure successfully completed.
```


4) Write a PL/SQL code to calculate purchase amount of customer where a

departmental store announces its festival offer to customers on cash payment. The

offer is as follows-

If purchase amount is less than 1000 then Tax=2% and discount=10%.

If purchase amount is greater than 1000 then Tax=5 % and discount=20%.

-->

```
SQL> DECLARE
  2     pa  NUMBER := &pa;
  3     fa  NUMBER;
  4  BEGIN
  5     IF pa < 1000 THEN
  6        fa := pa * 1.02 * 0.90;
  7     ELSE
  8        fa := pa * 1.05 * 0.80;
  9     END IF;
 10
 11     DBMS_OUTPUT.PUT_LINE('Purchase Amount: Rs. ' || pa);
 12     DBMS_OUTPUT.PUT_LINE('Final Amount: Rs. ' || fa);
 13  END;
 14  /
Enter value for pa: 8000
Purchase Amount: Rs. 8000
Final Amount: Rs. 6720


PL/SQL procedure successfully completed.


SQL> /
Enter value for pa: 15000
Purchase Amount: Rs. 15000
```

Final Amount: Rs. 12600

PL/SQL procedure successfully completed.

5) Write a PL/SQL code to calculate a simple interest and compound interest.

-->

```
SQL> DECLARE
  2     p  NUMBER := &p;
  3     r  NUMBER := &r;
  4     t  NUMBER := &t;
  5     si NUMBER;
  6     ci NUMBER;
  7  BEGIN
  8     si := (p * r * t) / 100;
  9
 10     ci := p * POWER((1 + r / 100), t) - p;
 11
 12     DBMS_OUTPUT.PUT_LINE('Simple Interest: Rs. ' || si);
 13     DBMS_OUTPUT.PUT_LINE('Compound Interest: Rs. ' || ci);
 14  END;
 15  /
Enter value for p: 10000
Enter value for r: 5
Enter value for t: 3
```

Simple Interest: Rs. 1500

Compound Interest: Rs. 1576.25

PL/SQL procedure successfully completed.

6) Write a PL/SQL code to calculate the sum of 'n' number digits.

-->

```
SQL> DECLARE
  2     n NUMBER := &n;
  3     temp NUMBER;
  4     d NUMBER;
  5     s NUMBER := 0;
  6  BEGIN
  7     temp := n;
  8
  9     WHILE temp > 0 LOOP
 10        d := MOD(temp, 10);
 11        s := s + d;
 12        temp := FLOOR(temp / 10);
 13     END LOOP;
 14
 15     DBMS_OUTPUT.PUT_LINE('Number: ' || n);
 16     DBMS_OUTPUT.PUT_LINE('Sum of Digits: ' || s);
 17  END;
```

```
18 /
Enter value for n: 12345
Number: 12345
Sum of Digits: 15

PL/SQL procedure successfully completed.
```

7) Write a PL/SQL code to swap two number.

-->

```
SQL> DECLARE
  2     a INT := &a;
  3     b INT := &b;
  4     temp INT;
  5  BEGIN
  6     DBMS_OUTPUT.PUT_LINE('Before Swap: a = ' || a || ', b = ' || b);
  7
  8     temp := a;
  9     a := b;
 10     b := temp;
 11
 12     DBMS_OUTPUT.PUT_LINE('After Swap: a = ' || a || ', b = ' || b);
 13  END;
 14  /
Enter value for a: 10
```

```
Enter value for b: 20
Before Swap: a = 10, b = 20
After Swap: a = 20, b = 10


PL/SQL procedure successfully completed.
```

8) Write a PL/SQL Program for Prime Number

-->

```
SQL> DECLARE
  2     n NUMBER := &n;
  3  BEGIN
  4     FOR i IN 2..n-1 LOOP
  5        IF MOD(n, i) = 0 THEN
  6           DBMS_OUTPUT.PUT_LINE(n || ' is not a prime number');
  7           RETURN;
  8        END IF;
  9     END LOOP;
 10
 11     DBMS_OUTPUT.PUT_LINE(n || ' is a prime number');
 12  END;
 13  /
Enter value for n: 11
11 is a prime number
```

```
PL/SQL procedure successfully completed.

SQL> /
Enter value for n: 12
12 is not a prime number

PL/SQL procedure successfully completed.
```

9) Write a PL/SQL Program to reverse a given String.

-->

```
SQL> DECLARE
  2     str VARCHAR2(100) := '&str';
  3     rev_str VARCHAR2(100) := '';
  4     i NUMBER;
  5  BEGIN
  6     FOR i IN REVERSE 1..LENGTH(str) LOOP
  7        rev_str := rev_str || SUBSTR(str, i, 1);
  8     END LOOP;
  9
 10     DBMS_OUTPUT.PUT_LINE('Original String: ' || str);
 11     DBMS_OUTPUT.PUT_LINE('Reversed String: ' || rev_str);
 12  END;
 13  /
Enter value for str: ABC
```

Original String: ABC

Reversed String: CBA

PL/SQL procedure successfully completed.

10) Write a PL/SQL Program for Fibonacci Series

-->

```
SQL> DECLARE
  2     n NUMBER := &n;
  3     a NUMBER := 0;
  4     b NUMBER := 1;
  5     temp NUMBER;
  6     i NUMBER;
  7  BEGIN
  8     DBMS_OUTPUT.PUT_LINE('Fibonacci Series:');
  9
 10     FOR i IN 1..n LOOP
 11        DBMS_OUTPUT.PUT_LINE(a);
 12        temp := a + b;
 13        a := b;
 14        b := temp;
 15     END LOOP;
 16  END;
 17  /
```

```
Enter value for n: 10
Fibonacci Series:
0
1
1
2
3
5
8
13
21
34

PL/SQL procedure successfully completed.
```


11) Write a PL/SQL Program to Find Factorial of a Number

-->

```
SQL> DECLARE
  2     n NUMBER := &n;
  3     fact NUMBER := 1;
  4     i NUMBER;
  5  BEGIN
  6     FOR i IN 1..n LOOP
  7        fact := fact * i;
```

```
  8     END LOOP;
  9
 10     DBMS_OUTPUT.PUT_LINE('Factorial of ' || n || ' is: ' || fact);
 11  END;
 12  /
Enter value for n: 5
Factorial of 5 is: 120


PL/SQL procedure successfully completed.
```

12) Write a PL/SQL code to calculate the area & perimeter of the rectangle, and the

area & circumference of the circle.

-->

```
SQL> DECLARE
  2     length NUMBER := &length;
  3     width   NUMBER := &width;
  4     radius NUMBER := &radius;
  5     PI CONSTANT NUMBER := 3.14159;
  6  BEGIN
  7
  8     DBMS_OUTPUT.PUT_LINE('Length: ' || length || ', Width: ' || width);
  9     DBMS_OUTPUT.PUT_LINE('Area of Rectangle: ' || length * width);
 10     DBMS_OUTPUT.PUT_LINE('Perimeter of Rectangle: ' || 2 * (length + width));
```

```
11      DBMS_OUTPUT.PUT_LINE('------------------------');
12      DBMS_OUTPUT.PUT_LINE('Radius: ' || radius);
13      DBMS_OUTPUT.PUT_LINE('Area of Circle: ' || PI * radius * radius);
14      DBMS_OUTPUT.PUT_LINE('Circumference of Circle: ' || 2 * PI * radius);
15  END;
16  /
Enter value for length: 10
Enter value for width: 5
Enter value for radius: 7
Length: 10, Width: 5
Area of Rectangle: 50
Perimeter of Rectangle: 30
------------------------
Radius: 7
Area of Circle: 153.93791
Circumference of Circle: 43.98226

PL/SQL procedure successfully completed.
```


13) Write a PL/SQL program using case statement

1) Check entered number is Armstrong or not.

2) Check entered number is Prime or not.

3) Check entered number is Palindrome or not.

-->

```
SQL> DECLARE
  2    n NUMBER := &n;
  3    ch NUMBER := &ch;
  4    temp NUMBER;
  5    s NUMBER := 0;
  6    r NUMBER;
  7  BEGIN
  8    CASE ch
  9      WHEN 1 THEN
 10        temp := n;
 11        WHILE temp > 0 LOOP
 12          r := MOD(temp, 10);
 13          s := s + (r * r * r);
 14          temp := FLOOR(temp / 10);
 15        END LOOP;
 16
 17        IF s = n THEN
 18          DBMS_OUTPUT.PUT_LINE(n || ' is Armstrong number');
 19        ELSE
 20          DBMS_OUTPUT.PUT_LINE(n || ' is not Armstrong number');
 21        END IF;
 22
 23      WHEN 2 THEN
 24        FOR i IN 2..n-1 LOOP
 25          IF MOD(n, i) = 0 THEN
```

```
26                DBMS_OUTPUT.PUT_LINE(n || ' is not prime number');
27                RETURN;
28              END IF;
29            END LOOP;
30            DBMS_OUTPUT.PUT_LINE(n || ' is prime number');
31
32          WHEN 3 THEN
33            temp := n;
34            WHILE temp > 0 LOOP
35              r := MOD(temp, 10);
36              s := s * 10 + r;
37              temp := FLOOR(temp / 10);
38            END LOOP;
39
40            IF s = n THEN
41              DBMS_OUTPUT.PUT_LINE(n || ' is palindrome number');
42            ELSE
43              DBMS_OUTPUT.PUT_LINE(n || ' is not palindrome number');
44            END IF;
45
46          ELSE
47            DBMS_OUTPUT.PUT_LINE('Invalid choice. Please enter 1, 2, or 3.');
48      END CASE;
49  END;
50  /
Enter value for n: 153
```

Enter value for ch: 1

153 is Armstrong number

PL/SQL procedure successfully completed.

SQL> /

Enter value for n: 4

Enter value for ch: 2

4 is not prime number

PL/SQL procedure successfully completed.

SQL> /

Enter value for n: 121

Enter value for ch: 3

121 is palindrome number

PL/SQL procedure successfully completed.

14) Write a PL/SQL Program for Reverse of a Number

-->

```
SQL> DECLARE
  2     n INT := &n;
  3     r INT := 0;
```

```
  4     temp INT;
  5  BEGIN
  6     temp := n;
  7     WHILE temp > 0 LOOP
  8        r := r * 10 + MOD(temp, 10);
  9        temp := FLOOR(temp / 10);
 10     END LOOP;
 11     DBMS_OUTPUT.PUT_LINE('Reversed Number: ' || r);
 12  END;
 13  /
Enter value for n: 123
Reversed Number: 321
```

15) Write a PL/SQL Program to Print Patterns

```
   *
   ***
   *****
   *******
   *********
```

-->

```
SQL> DECLARE
```

```
 2     n INT := &n;
 3     i INT;
 4     j INT;
 5  BEGIN
 6     FOR i IN 1..n LOOP
 7        FOR j IN 1..n - i LOOP
 8           DBMS_OUTPUT.PUT(' ');
 9        END LOOP;
10        FOR j IN 1..2 * i - 1 LOOP
11           DBMS_OUTPUT.PUT('*');
12        END LOOP;
13
14        DBMS_OUTPUT.NEW_LINE;
15     END LOOP;
16  END;
17  /
Enter value for n: 5
*
***
*****
*******
*********


PL/SQL procedure successfully completed.
```

16) Write a procedure to insert record in student table.

-->

```
SQL> CREATE OR REPLACE PROCEDURE insert_record(
  2     r IN student.rollno%TYPE,
  3     n IN student.name%TYPE,
  4     a IN student.age%TYPE
  5  ) AS
  6  BEGIN
  7     INSERT INTO student (rollno, name, age)
  8     VALUES (r, n, a);
  9     DBMS_OUTPUT.PUT_LINE('Record inserted successfully.');
 10  END;
 11  /

Procedure created.



SQL> EXEC insert_record(101, 'Ram', 20);
Record inserted successfully.

PL/SQL procedure successfully completed.

SQL> SELECT * FROM student;

    ROLLNO NAME                 AGE
```

```
---------- -------------------- ----------
       101 Ram                  20

1 rows selected.
```

17) Write a procedure to update record in the Employee table.

-->

```
SQL> CREATE OR REPLACE PROCEDURE update_emp(
  2     i IN emp.id%TYPE,
  3     n IN emp.name%TYPE
  4  ) AS
  5  BEGIN
  6     UPDATE emp
  7     SET name = n
  8     WHERE id = i;
  9
 10     DBMS_OUTPUT.PUT_LINE('Record updated successfully');
 11  END;
 12  /

Procedure created.

SQL> EXEC update_emp(101, 'RAM');
```

Record updated successfully

PL/SQL procedure successfully completed.

SQL> SELECT * FROM emp;

```
        ID NAME                           SALARY
---------- ------------------------------ ----------
       101 RAM                            10000
```

1 rows selected.

18) Write a pl/sql stored procedure for passing empno as a parameter display employee number, salary, bonus and his final salary.

-->

```
SQL> CREATE OR REPLACE PROCEDURE display_emp_details(
  2     no IN NUMBER
  3  ) AS
  4     v_salary NUMBER;
  5     bonus NUMBER;
  6     final_salary NUMBER;
  7  BEGIN
  8     SELECT salary
```

```
 9     INTO v_salary
10     FROM emp
11     WHERE id = no;
12
13     bonus := v_salary * 0.10;
14     final_salary := v_salary + bonus;
15
16     DBMS_OUTPUT.PUT_LINE('Employee Number: ' || no);
17     DBMS_OUTPUT.PUT_LINE('Salary: ' || v_salary);
18     DBMS_OUTPUT.PUT_LINE('Bonus: ' || bonus);
19     DBMS_OUTPUT.PUT_LINE('Final Salary: ' || final_salary);
20 END;
21 /
```

```
Procedure created.

SQL> EXEC display_emp_details(101);
Employee Number: 101
Salary: 10000
Bonus: 1000
Final Salary: 11000

PL/SQL procedure successfully completed.
```

19) Write a pl/sql function for calculating bonus of employee salary.

-->

```
SQL> CREATE OR REPLACE FUNCTION calculate_bonus(
  2     salary IN NUMBER
  3  ) RETURN NUMBER
  4  AS
  5     bonus NUMBER;
  6  BEGIN
  7     bonus := salary * 0.10;
  8
  9     RETURN bonus;
 10  END;
 11  /
```



```
Function created.

SQL> DECLARE
  2     empid  NUMBER := &empid;
  3     v_salary NUMBER;
  4
  5  BEGIN
  6     SELECT salary
  7     INTO v_salary
  8     FROM emp
  9     WHERE id = empid;
```

```
10
11      DBMS_OUTPUT.PUT_LINE('Salary: ' || v_salary);
12      DBMS_OUTPUT.PUT_LINE('Bonus: ' || calculate_bonus(v_salary));
13  END;
14  /
Enter value for empid: 101
Salary: 10000
Bonus: 1000


PL/SQL procedure successfully completed.
```

20) Write a pl/sql procedure for passing empid and display name ,salary and bouns

(which is calculated in above function).

-->

```
SQL> CREATE OR REPLACE PROCEDURE display_emp(
  2     empid IN NUMBER
  3  ) AS
  4     n VARCHAR(20);
  5     s NUMBER;
  6  BEGIN
  7     SELECT name, salary
  8     INTO n, s
  9     FROM emp
```

```
10      WHERE id = empid;
11
12      DBMS_OUTPUT.PUT_LINE('Employee Name: ' || n);
13      DBMS_OUTPUT.PUT_LINE('Salary: ' || s);
14      DBMS_OUTPUT.PUT_LINE('Bonus: ' || calculate_bonus(s));
15  END;
16  /

Procedure created.

SQL> EXEC display_emp(101);
Employee Name: RAM
Salary: 10000
Bonus: 1000

PL/SQL procedure successfully completed.
```