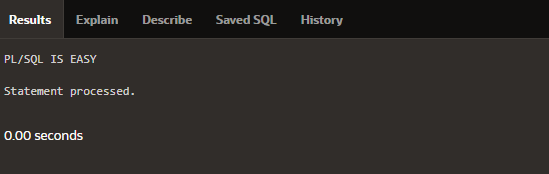
**PL/SQL**

* **USING BEGIN AND END:**

BEGIN

DBMS\_OUTPUT.PUT\_LINE('PL/SQL IS EASY');

END;



* **USING DECLARE:**

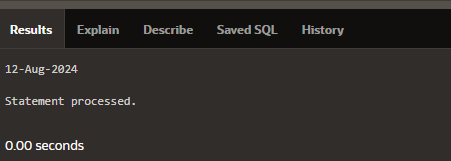
DECLARE

V\_DATE DATE:=SYSDATE;

BEGIN

DBMS\_OUTPUT.PUT\_LINE(V\_DATE);

END;



* **EXCEPTION:**

DECLARE

AGE NUMBER(10);

LAST\_NAME VARCHAR2(20);

BEGIN

SELECT AGE, LAST\_NAME

INTO AGE, LAST\_NAME

FROM COP3

WHERE LAST\_NAME = 'DIVYA';

DBMS\_OUTPUT.PUT\_LINE('EMPLOYEE OF THE MONTH IS: ' || AGE || ' ' || LAST\_NAME || '.');

EXCEPTION

WHEN TOO\_MANY\_ROWS THEN

DBMS\_OUTPUT.PUT\_LINE('Your SELECT statement retrieved multiple rows. Consider using a cursor or changing the search criteria.');

WHEN NO\_DATA\_FOUND THEN

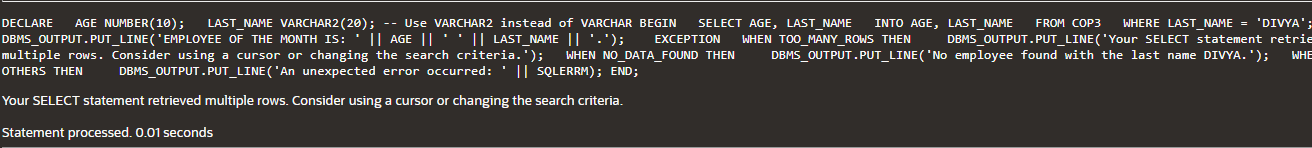
DBMS\_OUTPUT.PUT\_LINE('No employee found with the last name DIVYA.');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('An unexpected error occurred: ' || SQLERRM);

END;

/



* **ADDITION OF TWO NUMBER:**

DECLARE

a integer := 10;

b integer := 20;

c integer;

f real;

BEGIN

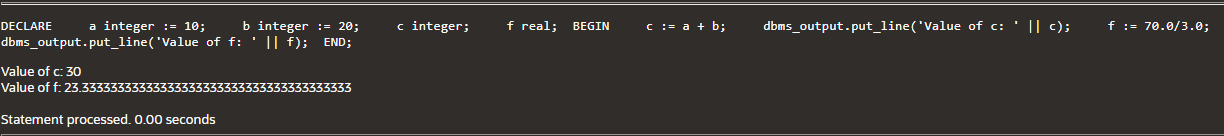
c := a + b;

dbms\_output.put\_line('Value of c: ' || c);

f := 70.0/3.0;

dbms\_output.put\_line('Value of f: ' || f);

END;



* **FIND THE RADIUS,DIAMETER,CIRCUMFRENEC AND AREA OF CIRCLE:**

DECLARE

-- constant declaration

pi constant number := 3.141592654;

-- other declarations

radius number(5,2);

dia number(5,2);

circumference number(7, 2);

area number (10, 2);

BEGIN

-- processing

radius := 9.5;

dia := radius \* 2;

circumference := 2.0 \* pi \* radius;

area := pi \* radius \* radius;

-- output

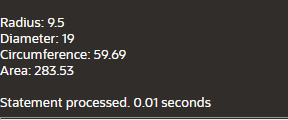
dbms\_output.put\_line('Radius: ' || radius);

dbms\_output.put\_line('Diameter: ' || dia);

dbms\_output.put\_line('Circumference: ' || circumference);

dbms\_output.put\_line('Area: ' || area);

END;



* **FIND THE GREATER NUMBER IN GIVEN TWO NUMBER:**

DECLARE

a INTEGER;

b INTEGER;

BEGIN

a := 10;

b := 20;

IF (a >= b) THEN

DBMS\_OUTPUT.PUT\_LINE('A IS GRETAER NUMBER');

ELSE

DBMS\_OUTPUT.PUT\_LINE('B IS GREATER NUMBER');

END IF;

END;



* **Count the word and character:**

DECLARE

str VARCHAR2(40) := 'Tutorials Point';

nchars NUMBER(4) := 0;

nwords NUMBER(4) := 1;

s CHAR(1);

BEGIN

FOR i IN 1..Length(str) LOOP

s := Substr(str, i, 1);

nchars := nchars + 1;

IF s = ' ' THEN

nwords := nwords + 1;

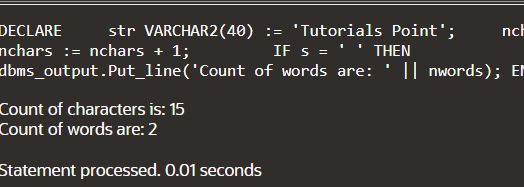
END IF;

END LOOP;

dbms\_output.Put\_line('Count of characters is: ' || nchars);

dbms\_output.Put\_line('Count of words are: ' || nwords);

END;

****

* **Sum of the number:**

DECLARE

sum\_of\_numbers NUMBER := 0;

BEGIN

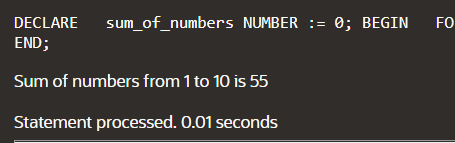
FOR i IN 1..10 LOOP

sum\_of\_numbers := sum\_of\_numbers + i;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('Sum of numbers from 1 to 10 is ' || sum\_of\_numbers);

END;



* **PRINT THR EVEN NUMBER:**

DECLARE

i NUMBER;

BEGIN

FOR i IN 1..20 LOOP

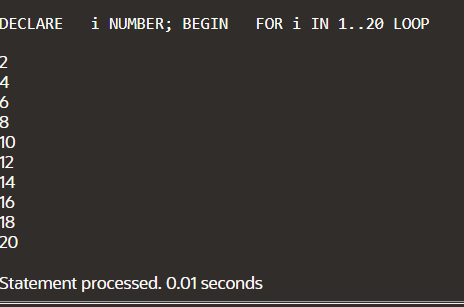
IF MOD(i, 2) = 0 THEN

DBMS\_OUTPUT.PUT\_LINE(i);

END IF;

END LOOP;

END;



* **ARRAY CONCEPT:**

DECLARE

type namesarray IS VARRAY(5) OF VARCHAR2(10);

type grades IS VARRAY(5) OF INTEGER;

names namesarray;

marks grades;

total integer;

BEGIN

names := namesarray('Kavita', 'Pritam', 'Ayan', 'Rishav', 'Aziz');

marks:= grades(98, 97, 78, 87, 92);

total := names.count;

dbms\_output.put\_line('Total '|| total || ' Students');

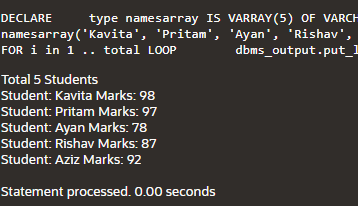
FOR i in 1 .. total LOOP

dbms\_output.put\_line('Student: ' || names(i) || '

Marks: ' || marks(i));

END LOOP;

END;



* **TO IDENTIFYING THE GRADE BASED ON MARKS:**

DECLARE

TYPE namesarray IS VARRAY(5) OF VARCHAR2(10);

TYPE grades IS VARRAY(5) OF INTEGER;

names namesarray;

marks grades;

total INTEGER;

BEGIN

names := namesarray('Kavita', 'Pritam', 'Ayan', 'Rishav', 'Aziz');

marks := grades(98, 97, 78, 87, 92);

total := names.count;

dbms\_output.put\_line('Total ' || total || ' Students');

FOR i IN 1 .. total LOOP

dbms\_output.put\_line('Student: ' || names(i) || ' Marks: ' || marks(i));

IF marks(i) > 95 THEN

dbms\_output.put\_line('S GRADE');

ELSIF marks(i) > 85 THEN

dbms\_output.put\_line('A GRADE');

ELSIF marks(i) > 75 THEN

dbms\_output.put\_line('B GRADE');

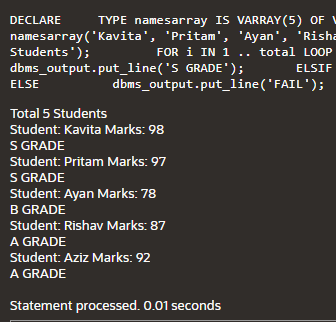
ELSE

dbms\_output.put\_line('FAIL');

END IF;

END LOOP;

END;



* **COUNT THE EVEN AND ODD NUMBER:**

GDECLARE

v\_start NUMBER := 1; -- Start of the range

v\_end NUMBER := 10; -- End of the range

v\_even\_count NUMBER := 0;

v\_odd\_count NUMBER := 0;

BEGIN

FOR i IN v\_start..v\_end LOOP

IF MOD(i, 2) = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('Even: ' || i);

v\_even\_count := v\_even\_count + 1;

ELSE

DBMS\_OUTPUT.PUT\_LINE('Odd: ' || i);

v\_odd\_count := v\_odd\_count + 1;

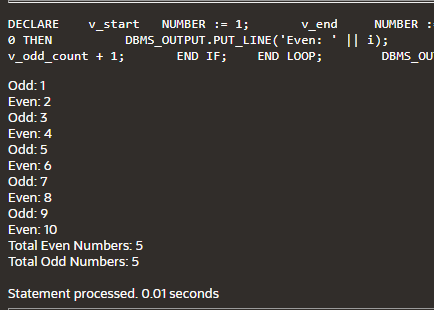
END IF;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('Total Even Numbers: ' || v\_even\_count);

DBMS\_OUTPUT.PUT\_LINE('Total Odd Numbers: ' || v\_odd\_count);

END;



* **COUNT THE EVEN AND ODD NUMBER IN GIVEN ARRAY:**

DECLARE

TYPE V\_NUMBER IS VARRAY(8) OF INTEGER(10);

v\_numbers V\_NUMBER := V\_NUMBER(2, 3, 1, 4, 6, 5, 7, 9);

v\_even\_count NUMBER := 0;

v\_odd\_count NUMBER := 0;

BEGIN

FOR i IN 1..v\_numbers.COUNT LOOP

IF MOD(v\_numbers(i), 2) = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('Even: ' || v\_numbers(i));

v\_even\_count := v\_even\_count + 1;

ELSE

DBMS\_OUTPUT.PUT\_LINE('Odd: ' || v\_numbers(i));

v\_odd\_count := v\_odd\_count + 1;

END IF;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('Total Even Numbers: ' || v\_even\_count);

DBMS\_OUTPUT.PUT\_LINE('Total Odd Numbers: ' || v\_odd\_count);

END;

* **COUNT THE EVEN AND ODD NUMBBER WITHOUT USING ARRAY FUNCTION:**

DECLARE

v\_number NUMBER := 123456789;

v\_digit NUMBER;

v\_num NUMBER;

v\_even\_count NUMBER := 0;

v\_odd\_count NUMBER := 0;

BEGIN

v\_num := v\_number;

WHILE v\_num > 0 LOOP

v\_digit := MOD(v\_num, 10);

IF MOD(v\_digit, 2) = 0 THEN

v\_even\_count := v\_even\_count + 1;

ELSE

v\_odd\_count := v\_odd\_count + 1;

END IF;

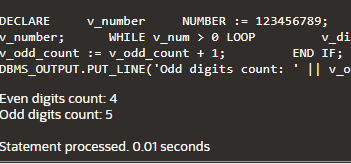
v\_num := FLOOR(v\_num / 10);

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('Even digits count: ' || v\_even\_count);

DBMS\_OUTPUT.PUT\_LINE('Odd digits count: ' || v\_odd\_count);

END;



* **PROCEDURE:**
* **FIND THE MINIMUM NUMBER BETWEEN TWO NUMBER:**

DECLARE

a number;

b number;

c number;

PROCEDURE findMin(x IN number, y IN number, z OUT number) IS

BEGIN

IF x < y THEN

z:= x;

ELSE

z:= y;

END IF;

END;

BEGIN

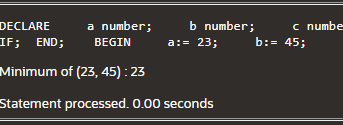
a:= 23;

b:= 45;

findMin(a, b, c);

G dbms\_output.put\_line(' Minimum of (23, 45) : ' || c);

END;



* **CALCULATOR WITHOUT USING FUNCTION:**

DECLARE

PROCEDURE calculator(p\_num1 IN NUMBER, p\_num2 IN NUMBER, p\_operation IN CHAR) IS

v\_result NUMBER;

BEGIN

IF p\_operation = '+' THEN

v\_result := p\_num1 + p\_num2;

ELSIF p\_operation = '-' THEN

v\_result := p\_num1 - p\_num2;

ELSIF p\_operation = '\*' THEN

v\_result := p\_num1 \* p\_num2;

ELSIF p\_operation = '/' THEN

IF p\_num2 <> 0 THEN

v\_result := p\_num1 / p\_num2;

ELSE

DBMS\_OUTPUT.PUT\_LINE('Error: Division by zero');

RETURN;

END IF;

ELSE

DBMS\_OUTPUT.PUT\_LINE('Error: Invalid operation');

RETURN;

END IF;

DBMS\_OUTPUT.PUT\_LINE('Result: ' || v\_result);

END;

BEGIN

calculator(10, 5, '+');

calculator(10, 5, '-');

calculator(10, 5, '\*');

calculator(10, 5, '/');

calculator(10, 0, '/');

calculator(10, 5, '%');

END;

* **CALCULATOR WITH USING SWITCH CASE:**

DECLARE

p\_num1 number;

p\_num2 number;

v\_result number;

PROCEDURE calculator(p\_num1 IN NUMBER, p\_num2 IN NUMBER, p\_operation IN CHAR) IS

v\_result NUMBER;

BEGIN

CASE p\_operation

WHEN '+' THEN

v\_result := p\_num1 + p\_num2;

WHEN '-' THEN

v\_result := p\_num1 - p\_num2;

WHEN '\*' THEN

v\_result := p\_num1 \* p\_num2;

WHEN '/' THEN

IF p\_num2 <> 0 THEN

v\_result := p\_num1 / p\_num2;

ELSE

DBMS\_OUTPUT.PUT\_LINE('Error: Division by zero');

RETURN;

END IF;

ELSE

DBMS\_OUTPUT.PUT\_LINE('Error: Invalid operation');

RETURN;

END CASE;

DBMS\_OUTPUT.PUT\_LINE('Result: ' || v\_result);

END;

BEGIN

calculator(10, 5, '+');

calculator(10, 5, '-');

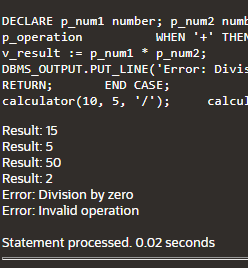
calculator(10, 5, '\*');

calculator(10, 5, '/');

calculator(10, 0, '/');

calculator(10, 5, '%');

END;



* **FACTORIAL USING FUNCTION:**

DECLARE

num number;

factorial number;

FUNCTION fact(x number)

RETURN number

IS

f number;

BEGIN

IF x=0 THEN

f := 1;

ELSE

f := x \* fact(x-1);

END IF;

RETURN f;

END;

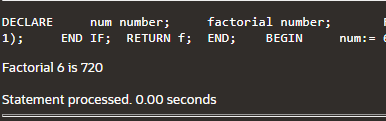
BEGIN

num:= 6;

factorial := fact(num);

dbms\_output.put\_line(' Factorial '|| num || ' is ' || factorial);

END;



* **TO FIND THE MAXIMUM NUMBER IN GIVEN NUMBER WITH USING FUNCTION:**

DECLARE

a number;

b number;

c number;

FUNCTION findMax(x IN number, y IN number)

RETURN number

IS

z number;

BEGIN

IF x > y THEN

z:= x;

ELSE

Z:= y;

END IF;

RETURN z;

END;

BEGIN

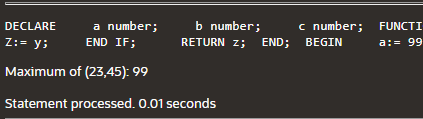
a:= 99;

b:= 23;

c := findMax(a, b);

dbms\_output.put\_line(' Maximum of (23,45): ' || c);

END;

\

* **FIBONACCISERIES:**

DECLARE

i NUMBER;

FUNCTION fibonacci(n NUMBER) RETURN NUMBER

IS

BEGIN

IF n <= 0 THEN

RETURN 0;

ELSIF n = 1 THEN

RETURN 1;

ELSE

RETURN fibonacci(n-1) + fibonacci(n-2);

END IF;

END fibonacci; -- add this to end the function declaration

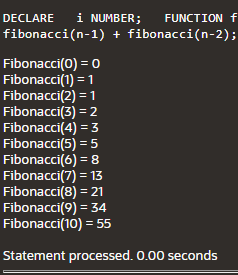
BEGIN

FOR i IN 0..10 LOOP

DBMS\_OUTPUT.PUT\_LINE('Fibonacci(' || i || ') = ' || fibonacci(i));

END LOOP;

END;



* **IMPLICITY:**

DECLARE

total\_rows number(2);

BEGIN

UPDATE COP3

SET AGE = AGE + 5;

IF sql%notfound THEN

dbms\_output.put\_line('no customers selected');

ELSIF sql%found THEN

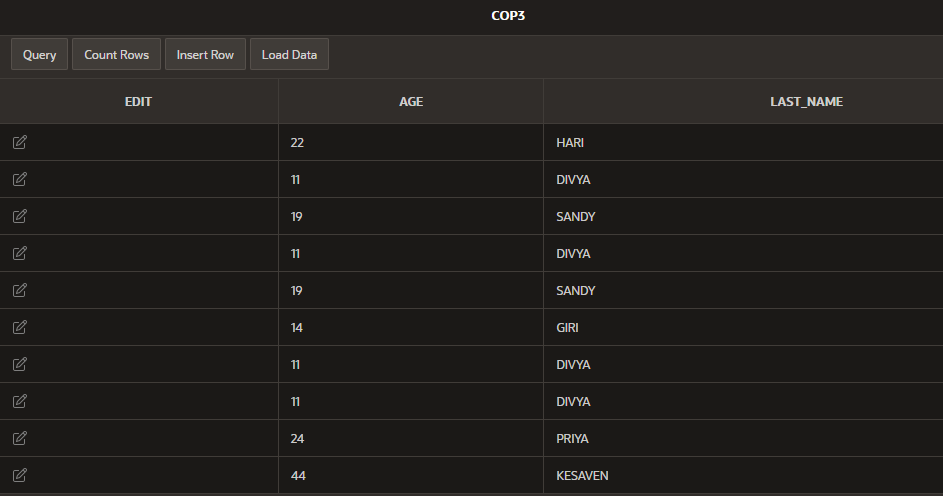
total\_rows := sql%rowcount;

dbms\_output.put\_line( total\_rows || ' customers selected ');

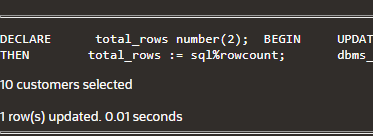
END IF;

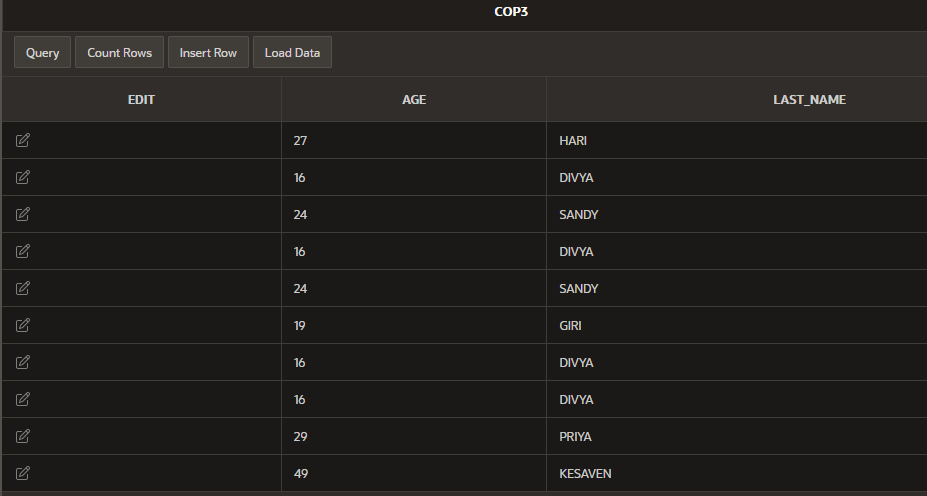
END;

**BEFORE TABLE:**



**OUTPUT:**





* **EXPLICITE:**

DECLARE

ID (link unavailable)%TYPE;

L\_NAME DD.L\_NAME%TYPE;

CURSOR DD\_CUR IS

SELECT ID, L\_NAME

FROM DD;

BEGIN

OPEN DD\_CUR;

LOOP

FETCH DD\_CUR INTO ID, L\_NAME;

EXIT WHEN DD\_CUR%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE(ID || ' ' || L\_NAME);

END LOOP;

CLOSE DD\_CUR;

END;