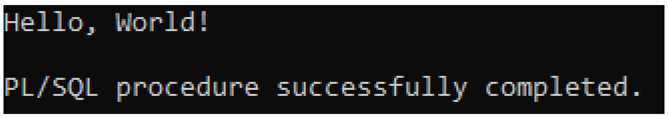
ANUBHAV SINGH

IC2K1654

MCA 8TH SEM

1. Hello world

```
DECLARE
message varchar2(20):= 'Hello, World!';
BEGIN
dbms_output.put_line(message);
end;
/
```

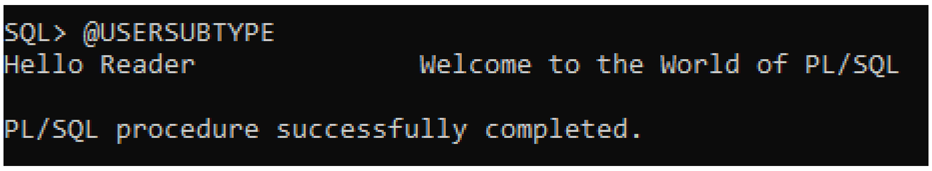


```
Hello, World!

PL/SQL procedure successfully completed.
```

2. User Subtype

```
DECLARE
SUBTYPE name IS char(20);
SUBTYPE message IS varchar2(100);
salutation name;
greetings message;
BEGIN salutation := 'Reader ';
greetings := 'Welcome to the World of PL/SQL';
dbms_output.put_line('Hello '||salutation||greetings);
END;
/
```

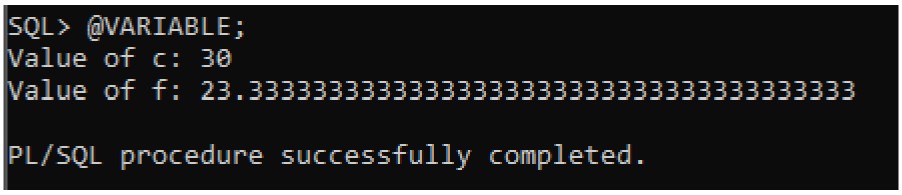


```
SQL> @USERSUBTYPE
Hello Reader              Welcome to the World of PL/SQL

PL/SQL procedure successfully completed.
```

3. Initializing variables in PL/SQL

```
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;
/
```

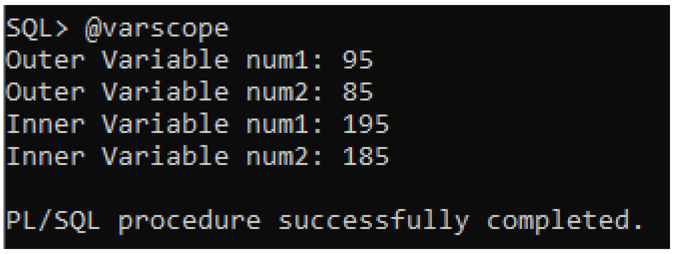


```
SQL> @VARIABLE;
Value of c: 30
Value of f: 23.33333333333333333333333333333333333333

PL/SQL procedure successfully completed.
```

4. **Scope of variables.**

```
DECLARE
num1 number := 95;
num2 number := 85;
BEGIN
dbms_output.put_line('Outer Variable num1: ' || num1);
dbms_output.put_line('Outer Variable num2: ' || num2);
DECLARE
num1 number := 195;
num2 number := 185;
BEGIN
dbms_output.put_line('Inner Variable num1: ' || num1);
dbms_output.put_line('Inner Variable num2: ' || num2);
END;
END;
/
```

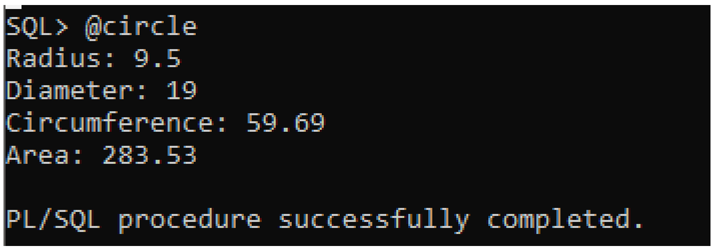


```
SQL> @varscope
Outer Variable num1: 95
Outer Variable num2: 85
Inner Variable num1: 195
Inner Variable num2: 185

PL/SQL procedure successfully completed.
```

5. **Declaration of constant using measurements of circle.**

```
DECLARE
pi constant number := 3.141592654;
radius number(5,2);
dia number(5,2);
circumference number(7, 2);
area number (10, 2);
BEGIN
radius := 9.5;
dia := radius * 2;
circumference := 2.0 * pi * radius;
area := pi * radius * radius;
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;
/
```

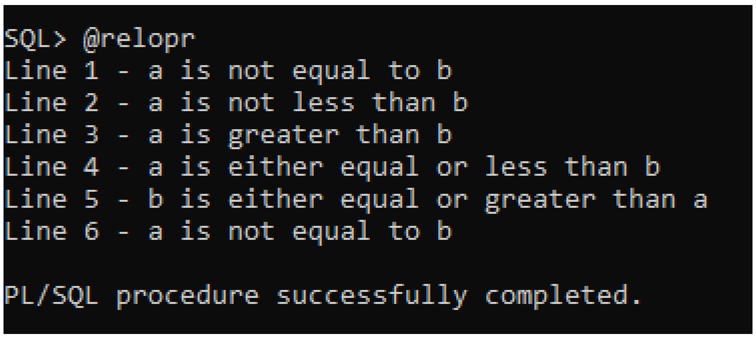


```
SQL> @circle
Radius: 9.5
Diameter: 19
Circumference: 59.69
Area: 283.53

PL/SQL procedure successfully completed.
```

6. Relational operator.

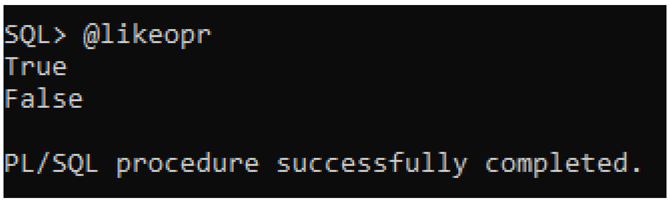
```
DECLARE
a number (2) := 21;
b number (2) := 10;
BEGIN
IF (a = b)
then
dbms_output.put_line('Line 1 - a is equal to b');
ELSE
dbms_output.put_line('Line 1 - a is not equal to b');
END IF;
IF (a < b)
then
dbms_output.put_line('Line 2 - a is less than b');
ELSE
dbms_output.put_line('Line 2 - a is not less than b');
END IF;
IF ( a > b )
THEN
dbms_output.put_line('Line 3 - a is greater than b');
ELSE
dbms_output.put_line('Line 3 - a is not greater than b');
END IF;
a := 5;
b := 20;
IF ( a <= b )
THEN
dbms_output.put_line('Line 4 - a is either equal or less than b');
END IF;
IF ( b >= a )
THEN
dbms_output.put_line('Line 5 - b is either equal or greater than a');
END IF;
IF ( a <> b )
THEN dbms_output.put_line('Line 6 - a is not equal to b');
ELSE
dbms_output.put_line('Line 6 - a is equal to b');
END IF;
END;
/
```


```
SQL> @relopr
Line 1 - a is not equal to b
Line 2 - a is not less than b
Line 3 - a is greater than b
Line 4 - a is either equal or less than b
Line 5 - b is either equal or greater than a
Line 6 - a is not equal to b

PL/SQL procedure successfully completed.
```

7. Like operator.

```
DECLARE
PROCEDURE compare (value varchar2, pattern varchar2 ) is
BEGIN
IF value LIKE pattern
THEN
dbms_output.put_line ('True');
ELSE
dbms_output.put_line ('False');
END IF;
END;
BEGIN
compare('Zara Ali', 'Z%A_i');
compare('Nuha Ali', 'Z%A_i');
END;
/
```

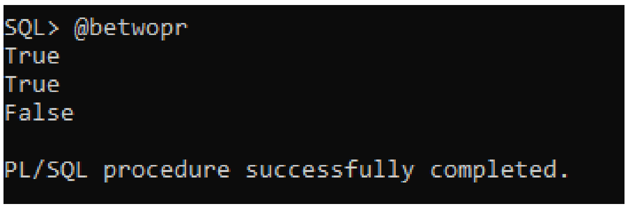

```
SQL> @likeopr
True
False

PL/SQL procedure successfully completed.
```

8. Between operator

```
DECLARE
x number(2) := 10;
BEGIN
IF (x between 5 and 20)
THEN
dbms_output.put_line('True');
ELSE
dbms_output.put_line('False');
END IF;
IF (x BETWEEN 5 AND 10)
THEN
dbms_output.put_line('True');
ELSE
dbms_output.put_line('False');
END IF;
IF (x BETWEEN 11 AND 20)
```

```
THEN
dbms_output.put_line('True');
ELSE
dbms_output.put_line('False');
END IF;
END;
/
```

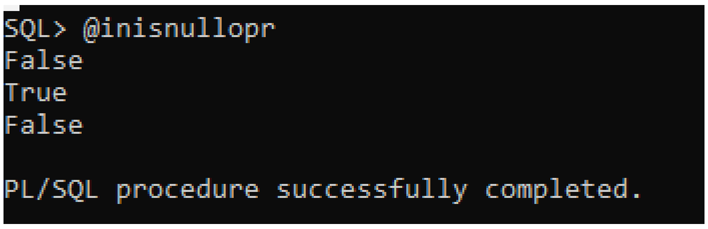


```
SQL> @betwopr
True
True
False

PL/SQL procedure successfully completed.
```

9. In and Is Null operator

```
DECLARE
letter varchar2(1) := 'm';
BEGIN
IF (letter in ('a', 'b', 'c'))
THEN
dbms_output.put_line('True');
ELSE
dbms_output.put_line('False');
END IF;
IF (letter in ('m', 'n', 'o'))
THEN
dbms_output.put_line('True');
ELSE
dbms_output.put_line('False');
END IF;
IF (letter is null)
THEN
dbms_output.put_line('True');
ELSE
dbms_output.put_line('False');
END IF;
END;
/
```



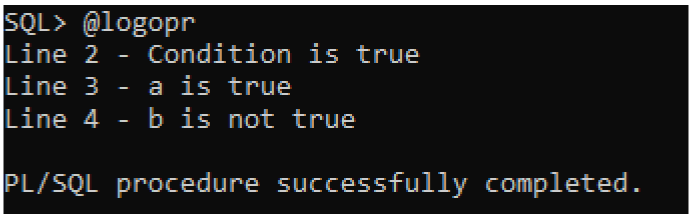
```
SQL> @inisnullopr
False
True
False

PL/SQL procedure successfully completed.
```

10. Logical operator.

```
DECLARE
a boolean := true;
b boolean := false;
BEGIN
IF (a AND b)
THEN
dbms_output.put_line('Line 1 - Condition is true');
```

```
END IF;
IF (a OR b)
THEN
dbms_output.put_line('Line 2 - Condition is true');
END IF;
IF (NOT a)
THEN
dbms_output.put_line('Line 3 - a is not true');
ELSE
dbms_output.put_line('Line 3 - a is true');
END IF;
IF (NOT b)
THEN
dbms_output.put_line('Line 4 - b is not true');
ELSE
dbms_output.put_line('Line 4 - b is true');
END IF;
END;
/
```

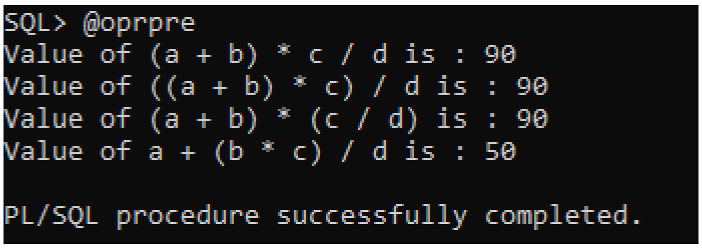


```
SQL> @logopr
Line 2 - Condition is true
Line 3 - a is true
Line 4 - b is not true

PL/SQL procedure successfully completed.
```

11. Operator precedence.

```
DECLARE
a number(2) := 20;
b number(2) := 10;
c number(2) := 15;
d number(2) := 5;
e number(2) ;
BEGIN
e := (a + b) * c / d;
dbms_output.put_line('Value of (a + b) * c / d is : '|| e );
e := ((a + b) * c) / d;
dbms_output.put_line('Value of ((a + b) * c) / d is : ' || e );
e := (a + b) * (c / d);
dbms_output.put_line('Value of (a + b) * (c / d) is : '|| e );
e := a + (b * c) / d;
dbms_output.put_line('Value of a + (b * c) / d is : ' || e );
END;
/
```

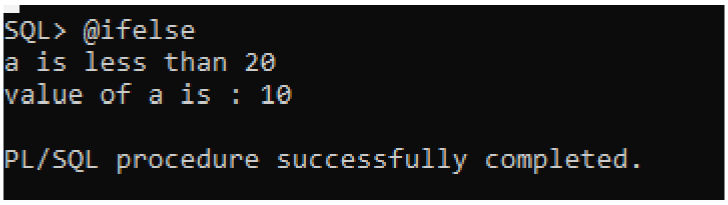


```
SQL> @oprpre
Value of (a + b) * c / d is : 90
Value of ((a + b) * c) / d is : 90
Value of (a + b) * (c / d) is : 90
Value of a + (b * c) / d is : 50

PL/SQL procedure successfully completed.
```

12. IF ELSE.

```
DECLARE
a number(2) := 10;
BEGIN
a:= 10;
IF( a < 20 )
THEN
dbms_output.put_line('a is less than 20 ' );
END IF;
dbms_output.put_line('value of a is : ' || a);
END;
/
```

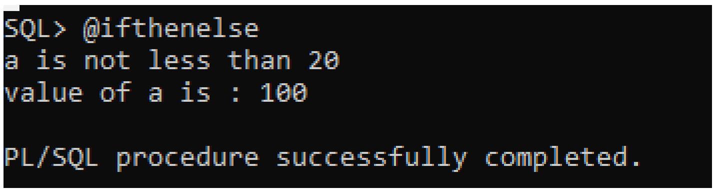


```
SQL> @ifelse
a is less than 20
value of a is : 10

PL/SQL procedure successfully completed.
```

13. IF THEN ELSE

```
DECLARE
a number(3) := 100;
BEGIN
IF( a < 20 )
THEN
dbms_output.put_line('a is less than 20 ' );
ELSE
dbms_output.put_line('a is not less than 20 ' );
END IF;
dbms_output.put_line('value of a is : ' || a);
END;
/
```



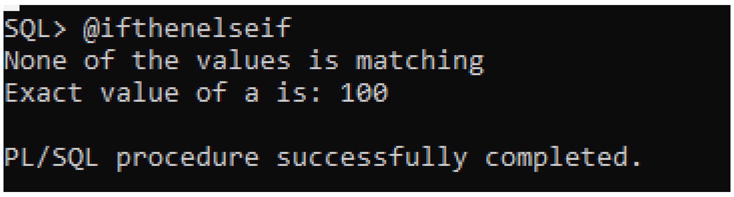
```
SQL> @ifthenelse
a is not less than 20
value of a is : 100

PL/SQL procedure successfully completed.
```

14. IF THEN ELSEIF

```
DECLARE
a number(3) := 100;
BEGIN
IF ( a = 10 )
THEN
dbms_output.put_line('Value of a is 10' );
ELSIF ( a = 20 )
THEN
dbms_output.put_line('Value of a is 20' );
ELSIF ( a = 30 )
THEN
dbms_output.put_line('Value of a is 30' );
ELSE
```

```
dbms_output.put_line('None of the values is matching');
END IF;
dbms_output.put_line('Exact value of a is: '|| a );
END;
/
```

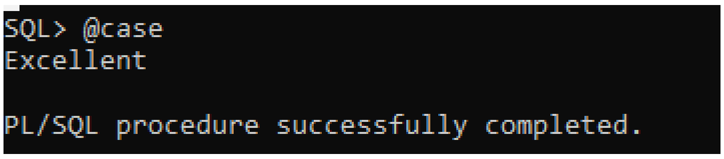


```
SQL> @ifthenelseif
None of the values is matching
Exact value of a is: 100

PL/SQL procedure successfully completed.
```

15. CASE Statement.

```
DECLARE
grade char(1) := 'A';
BEGIN
CASE grade
when 'A' then
dbms_output.put_line('Excellent');
when 'B' then
dbms_output.put_line('Very good');
when 'C' then
dbms_output.put_line('Well done');
when 'D' then
dbms_output.put_line('You passed');
when 'F' then
dbms_output.put_line('Better try again');
else
dbms_output.put_line('No such grade');
END CASE;
END;
/
```



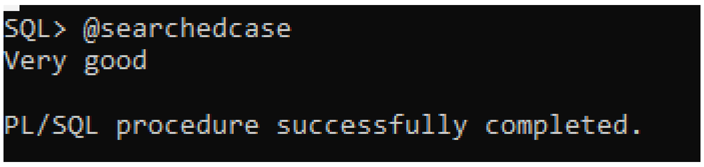
```
SQL> @case
Excellent

PL/SQL procedure successfully completed.
```

16. Searched CASE Statement.

```
DECLARE
grade char(1) := 'B';
BEGIN
case
when grade = 'A'
then
dbms_output.put_line('Excellent');
when grade = 'B'
then
dbms_output.put_line('Very good');
when grade = 'C'
then
dbms_output.put_line('Well done');
```

```
when grade = 'D'
then
dbms_output.put_line('You passed');
when grade = 'F' then
dbms_output.put_line('Better try again');
else
dbms_output.put_line('No such grade');
end case;
end;
/
```

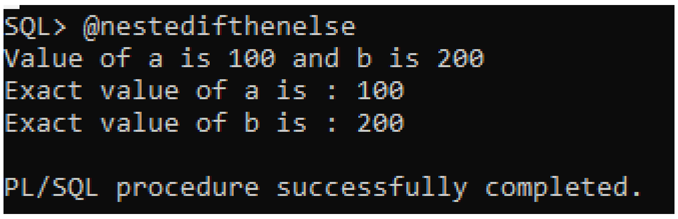


```
SQL> @searchedcase
Very good

PL/SQL procedure successfully completed.
```

17. Nested IF THEN ELSE

```
DECLARE
a number(3) := 100;
b number(3) := 200;
BEGIN
IF( a = 100 )
THEN
IF( b = 200 )
THEN
dbms_output.put_line('Value of a is 100 and b is 200' );
END IF;
END IF;
dbms_output.put_line('Exact value of a is : ' || a );
dbms_output.put_line('Exact value of b is : ' || b );
END;
/
```

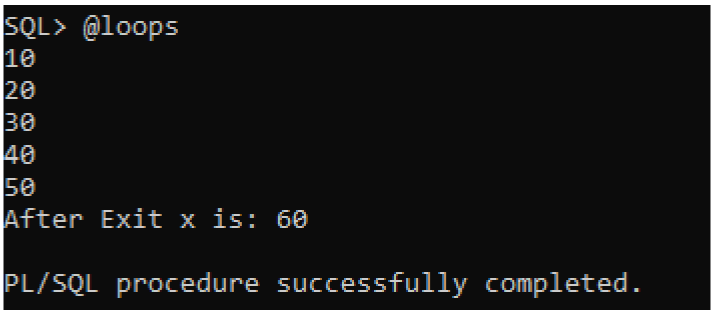


```
SQL> @nestedifthenelse
Value of a is 100 and b is 200
Exact value of a is : 100
Exact value of b is : 200

PL/SQL procedure successfully completed.
```

18. LOOPS

```
DECLARE
x number := 10;
BEGIN
LOOP
dbms_output.put_line(x);
x := x + 10;
IF x > 50
THEN exit;
END IF;
END LOOP;
dbms_output.put_line('After Exit x is: ' || x);
END;
/
```

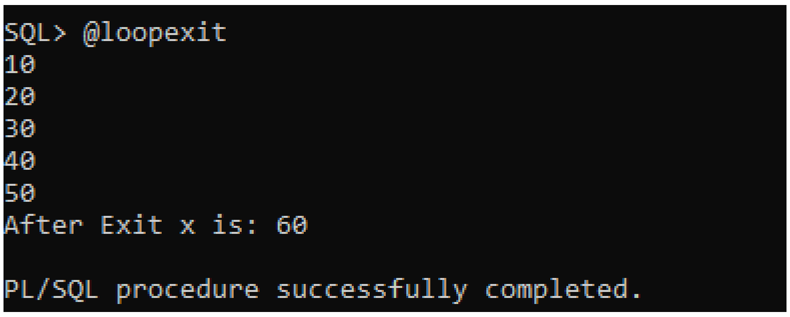


```
SQL> @loops
10
20
30
40
50
After Exit x is: 60

PL/SQL procedure successfully completed.
```

19. LOOP EXIT WHEN Statement

```
DECLARE
x number := 10;
BEGIN
LOOP
dbms_output.put_line(x);
x := x + 10;
exit
WHEN x > 50;
END LOOP;
dbms_output.put_line('After Exit x is: ' || x);
END;
/
```

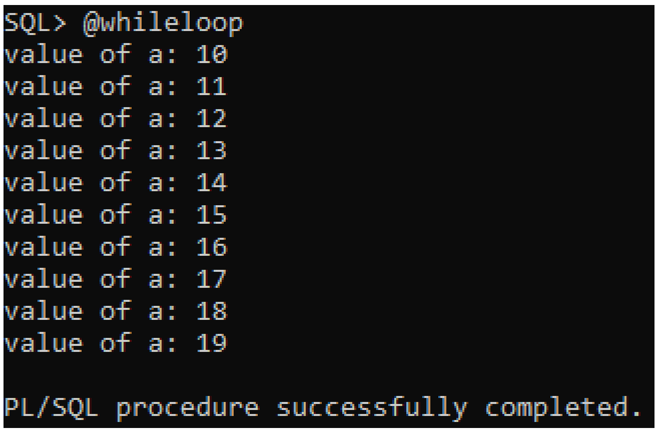


```
SQL> @loopexit
10
20
30
40
50
After Exit x is: 60

PL/SQL procedure successfully completed.
```

20. WHILE LOOP

```
DECLARE
a number(2) := 10;
BEGIN
WHILE a < 20 LOOP
dbms_output.put_line('value of a: ' || a);
a := a + 1;
END LOOP;
END;
/
```

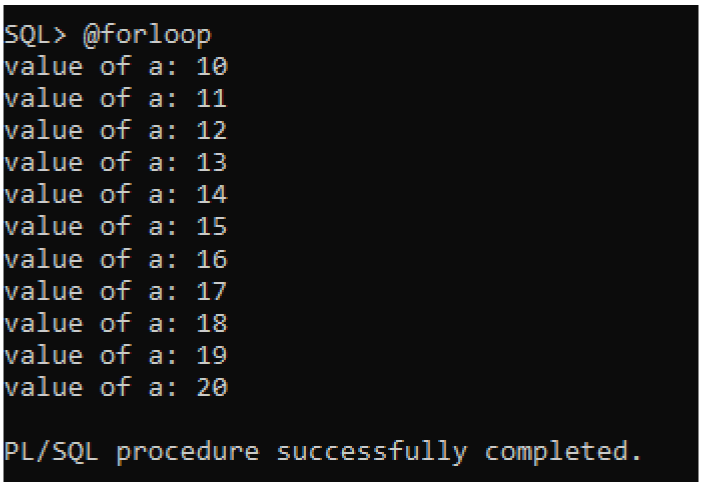


```
SQL> @whileloop
value of a: 10
value of a: 11
value of a: 12
value of a: 13
value of a: 14
value of a: 15
value of a: 16
value of a: 17
value of a: 18
value of a: 19

PL/SQL procedure successfully completed.
```

21. FOR LOOP.

```
DECLARE
a number(2);
BEGIN
FOR a in 10 .. 20 LOOP
dbms_output.put_line('value of a: ' || a);
END LOOP;
END;
/
```

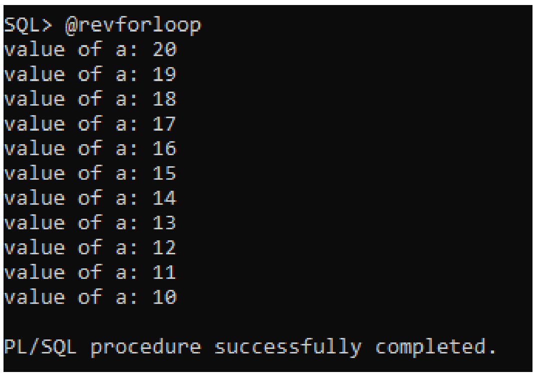


```
SQL> @forloop
value of a: 10
value of a: 11
value of a: 12
value of a: 13
value of a: 14
value of a: 15
value of a: 16
value of a: 17
value of a: 18
value of a: 19
value of a: 20

PL/SQL procedure successfully completed.
```

22. REVERSE FOR LOOP.

```
DECLARE
a number(2) ;
BEGIN
FOR a IN REVERSE 10 .. 20 LOOP
dbms_output.put_line('value of a: ' || a);
END LOOP;
END;
/
```

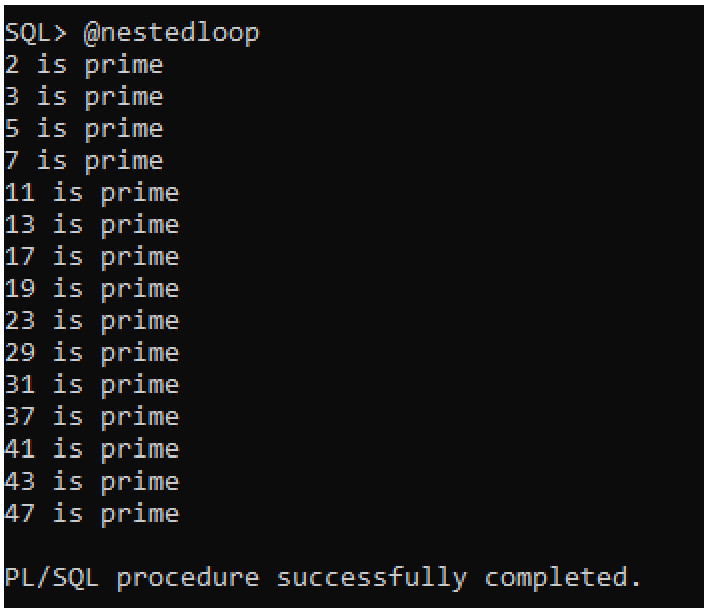


```
SQL> @revforloop
value of a: 20
value of a: 19
value of a: 18
value of a: 17
value of a: 16
value of a: 15
value of a: 14
value of a: 13
value of a: 12
value of a: 11
value of a: 10

PL/SQL procedure successfully completed.
```

23. NESTED LOOPS

```
DECLARE
i number(3);
j number(3);
BEGIN
i := 2;
LOOP j:= 2;
LOOP exit WHEN ((mod(i, j) = 0) or (j = i));
j := j +1;
END LOOP;
IF (j = i )
THEN
dbms_output.put_line(i || ' is prime');
END IF;
i := i + 1;
exit WHEN i = 50;
END LOOP;
END;
/
```



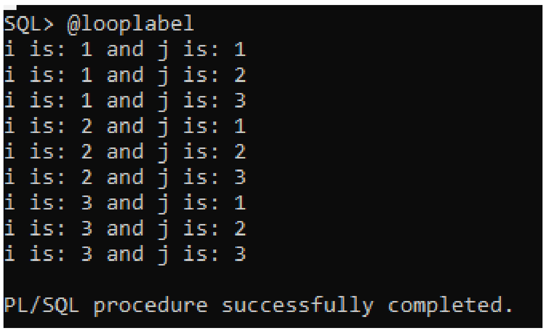
```
SQL> @nestedloop
2 is prime
3 is prime
5 is prime
7 is prime
11 is prime
13 is prime
17 is prime
19 is prime
23 is prime
29 is prime
31 is prime
37 is prime
41 is prime
43 is prime
47 is prime

PL/SQL procedure successfully completed.
```

24. LABELING A LOOP

```
DECLARE
i number(1);
```

j number(1);
BEGIN
<< outer_loop >>
FOR i IN 1..3 LOOP
<< inner_loop >>
FOR j IN 1..3 LOOP
dbms_output.put_line('i is: '|| i || ' and j is: ' || j);
END loop inner_loop;
END loop outer_loop;
END;
/

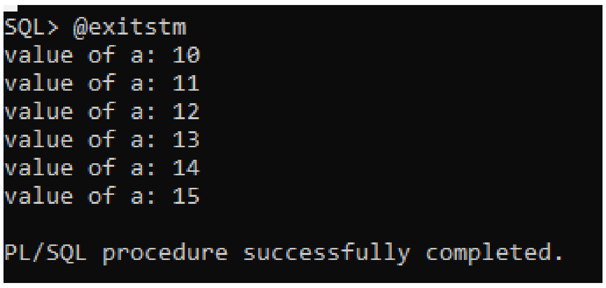


```
SQL> @looplabel
i is: 1 and j is: 1
i is: 1 and j is: 2
i is: 1 and j is: 3
i is: 2 and j is: 1
i is: 2 and j is: 2
i is: 2 and j is: 3
i is: 3 and j is: 1
i is: 3 and j is: 2
i is: 3 and j is: 3

PL/SQL procedure successfully completed.
```

25. Exit Statement

DECLARE
a number(2) := 10;
BEGIN
WHILE a < 20 LOOP
dbms_output.put_line ('value of a: ' || a);
a := a + 1;
IF a > 15 THEN
EXIT;
END IF;
 END LOOP;
END;
/



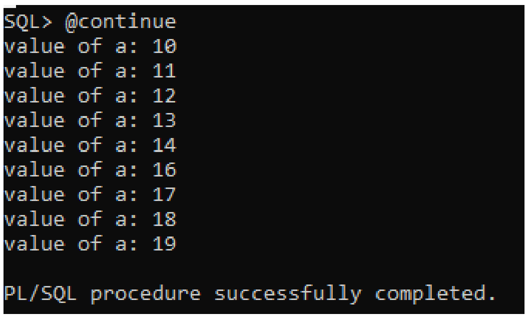
```
SQL> @exitstm
value of a: 10
value of a: 11
value of a: 12
value of a: 13
value of a: 14
value of a: 15

PL/SQL procedure successfully completed.
```

26. CONTINUE Statement

DECLARE
a number(2) := 10;
BEGIN
WHILE a < 20 LOOP
dbms_output.put_line ('value of a: ' || a);
a := a + 1;
IF a = 15 THEN
a := a + 1;
CONTINUE;

```
END IF;
END LOOP;
END;
/
```

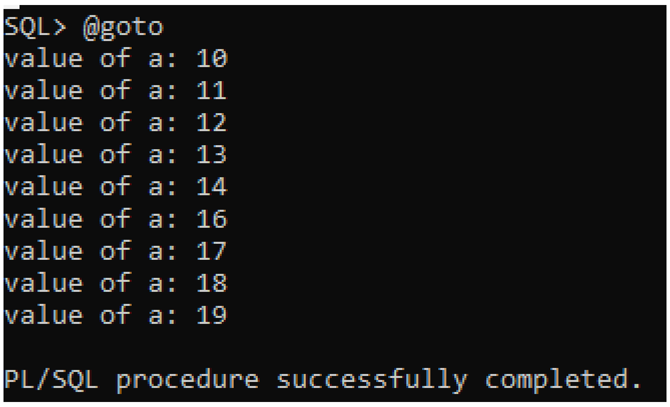


```
SQL> @continue
value of a: 10
value of a: 11
value of a: 12
value of a: 13
value of a: 14
value of a: 16
value of a: 17
value of a: 18
value of a: 19

PL/SQL procedure successfully completed.
```

27. GOTO Statement.

```
DECLARE
a number(2) := 10;
BEGIN
<<loopstart>>
WHILE a < 20 LOOP
dbms_output.put_line ('value of a: ' || a);
a := a + 1;
IF a = 15 THEN
a := a + 1;
GOTO loopstart;
END IF;
END LOOP;
END;
/
```



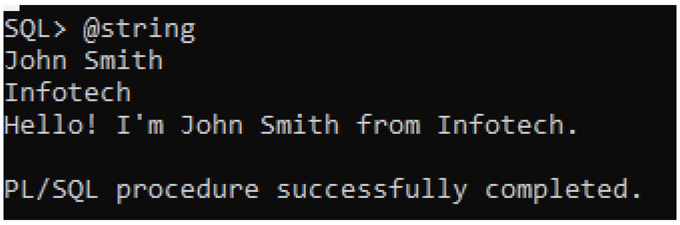
```
SQL> @goto
value of a: 10
value of a: 11
value of a: 12
value of a: 13
value of a: 14
value of a: 16
value of a: 17
value of a: 18
value of a: 19

PL/SQL procedure successfully completed.
```

28. String Variables

```
DECLARE
name varchar2(20);
company varchar2(30);
introduction clob;
```

```
choice char(1);
BEGIN
name := 'John Smith';
company := 'Infotech';
introduction := ' Hello! I''m John Smith from Infotech.';
choice := 'y';
IF choice = 'y' THEN
dbms_output.put_line(name);
dbms_output.put_line(company);
dbms_output.put_line(introduction);
END IF;
END;
/
```

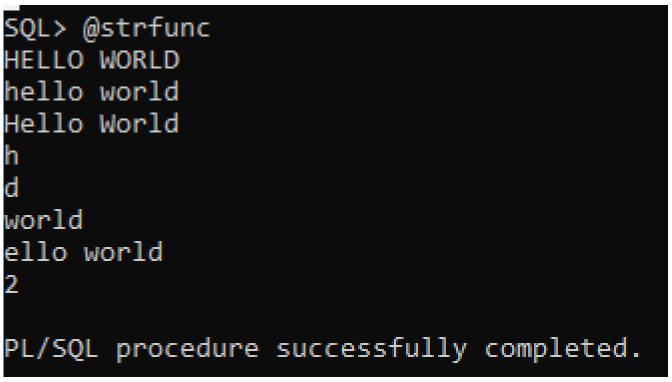


```
SQL> @string
John Smith
Infotech
Hello! I'm John Smith from Infotech.

PL/SQL procedure successfully completed.
```

29. String Functions

```
DECLARE
greetings varchar2(11) := 'hello world';
BEGIN
dbms_output.put_line(UPPER(greetings));
dbms_output.put_line(LOWER(greetings));
dbms_output.put_line(INITCAP(greetings));
dbms_output.put_line ( SUBSTR (greetings, 1, 1));
dbms_output.put_line ( SUBSTR (greetings, -1, 1));
dbms_output.put_line ( SUBSTR (greetings, 7, 5));
dbms_output.put_line ( SUBSTR (greetings, 2));
dbms_output.put_line ( INSTR (greetings, 'e'));
END;
/
```

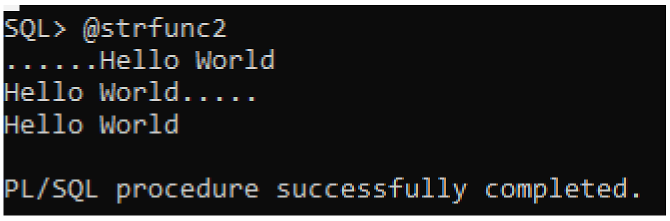


```
SQL> @strfunc
HELLO WORLD
hello world
Hello World
h
d
world
ello world
2

PL/SQL procedure successfully completed.
```

30. Trim in string

```
DECLARE
greetings varchar2(30) := '......Hello World.....';
BEGIN
dbms_output.put_line(RTRIM(greetings,'.'));
dbms_output.put_line(LTRIM(greetings, '.'));
dbms_output.put_line(TRIM( '.' from greetings));
END;
/
```



```
SQL> @strfunc2
......Hello World
Hello World.....
Hello World

PL/SQL procedure successfully completed.
```