SQL General Functions: NVL, NVL2, DECODE, COALESCE, NULLIF, LNNVL and NANVL

Introduction

Today, we are going to learn about SQL general Functions. The General Functions we are going to learn about are:

- NVL()
- NVL2()
- DECODE()
- COALESCE()
- LNNVL()

1.) NVL()

This is one of the functions of SQL extensively used in Structured Query Language (SQL).

This function can hold two input values only. If input values are more than 2 then an error is returned. This functions returns the first NOT NULL value when searched in the function.

If both the inputs are NULL, then there is no output for this function.

The input data type can be Integer, Floating Point Number, String, Character input, etc.

Syntax

```
1. NVL (input 1, input 2)
```

Example Queries

```
SQL> select NVL(1, 2) from dual;


NVL(1,2)

  1
```

```
SQL> select NVL(NULL, 1) from dual;


NVL(NULL,1)

_ _ _ _ _ _ _ _ _
        1
SQL> select NVL(1.029384, 1.029384) from dual;

NVL(1.029384,1.029384)

_ _ _ _ _ _ _ _ _ _ _ _ _ _ _


            1.029384

SQL> select NVL(NULL, 1.029384) from dual;

NVL(NULL,1.029384)

_ _ _ _ _ _ _ _ _ _ _ _ _
       1.029384

SQL> select NVL('JOE', 'ROOT') from dual;

NVL

_ _ _
JOE

SQL> select NVL(NULL, 'ROOT') from dual;

NVL(

_ _ _ _
ROOT
SQL> select COMMISION_PERCENTAGE from sal;

COMMISION_PERCENTAGE

_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
              .15
              .1
             .01
```

```
          .33
SQL> select NVL(COMMISION_PERCENTAGE, 0) from sal;

NVL(COMMISION_PERCENTAGE,0)
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
                .15
                 .1
                 .01
                  0
                   0
                .33
                   0
                   0

8 rows selected.
SQL> SELECT  id, name, sal, NVL (COMMISION_PERCENTAGE, 0),
2        (sal) + (sal * NVL (COMMISION_PERCENTAGE, 0))
3         monthly_salary FROM sal;

ID   NAME                  SAL     NVL(COMMISION_PERCENTAGE,0)   MONTHL
Y_SALARY
1. _ _   _ _ _ _ _ _ _ _      _ _ _    _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

1    Joe Root             75000            0.15                  86250
2    Ros Taylor            90000            0.1                   99000
3    Paul Adams            50000            0.01                  50500
4    Victor Lee            43000             0                    43000
5    Matt Potts             20000            0                    20000
6    James Anderson         200000           0.33                  266000
7    Craig Overton         11000            0                    11000
8    Rory Burns             9000            0                     9000

8 rows selected.
SQL> SELECT  id, name, sal, NVL (COMMISION_PERCENTAGE, 0),
2        (sal * 12) + (sal * NVL (COMMISION_PERCENTAGE, 0) *12)
```

```
 3          yearly_salary FROM sal;
ID   NAME                 SAL     NVL(COMMISION_PERCENTAGE,0)   YEARLY_
SALARY
--   ---------       ---    -------------------- -----------

1    Joe Root           75000          0.15                    1035000
2    Ros Taylor         90000          0.1                     1188000
3    Paul Adams         50000          0.01                    606000
4    Victor Lee         43000          0                       516000
5    Matt Potts         20000          0                       240000
6    James Anderson     200000         0.33                    3192000
7    Craig Overton      11000          0                       132000
8    Rory Burns         9000           0                       108000

8 rows selected.
```

2.) NVL2()

This is one of the functions of SQL extensively used in Structured Query Language (SQL).

This function can hold three input values only. If input values are more than three then an error is returned. This function returns the first value after NOT NULL value is found, when searched in the function.

If the first value is NOT NULL, second value is returned.

If the first value is NULL and the second value is NOT NULL, third value is returned.

The returned value can also be a NULL value too.

The working of this functioning is as same as NVL () in SQL.

If both the inputs are NULL, then there is no output for this function.

The input data type can be Integer, Floating Point Number, String, Character input, etc.

Syntax

```
NVL2(input 1, input 2, input 3)
```

Example Queries

```
SQL> select NVL2(1, 2, 3) from dual;

NVL2(1,2,3)
_ _ _ _ _ _ _ _ _
          2

SQL> select NVL2(2, 2, 3) from dual;

NVL2(2,2,3)
_ _ _ _ _ _ _ _ _
          2

SQL> select NVL2(2, 4, 3) from dual;

NVL2(2,4,3)
_ _ _ _ _ _ _ _ _
          4

SQL> select NVL2(2, NULL, 3) from dual;

NVL2(2,NULL,3)
_ _ _ _ _ _ _ _ _ _


SQL> select NVL2('Kevin', 'Pitersen', ' SA / ENG ') from dual;

NVL2('KE
_ _ _ _ _ _ _
Pitersen

SQL> select NVL2('NULL', 'Pitersen', ' SA / ENG ') from dual;

NVL2('NU
_ _ _ _ _ _
Pitersen

SQL> select NVL2(NULL, 'Pitersen', ' SA / ENG ') from dual;

NVL2(NULL,
_ _ _ _ _ _
 SA / ENG

SQL> select NVL2('Kevin', NULL, ' SA / ENG ') from dual;

N
_

SQL> select NVL2(56.2, 35.6, 23.4) from dual;

NVL2(56.2,35.6,23.4)
_ _ _ _ _ _ _ _ _ _ _ _
              35.6

SQL> select NVL2(NULL, 35.6, 23.4) from dual;

NVL2(NULL,35.6,23.4)
_ _ _ _ _ _ _ _ _ _ _ _ _
              23.4

SQL> select NVL2(NULL, NULL, 23.4) from dual;

NVL2(NULL,NULL,23.4)
_ _ _ _ _ _ _ _ _ _ _ _ _
              23.4

SQL> select NVL2(NULL, NULL, NULL) from dual;

N
_

```

```

SQL> select NVL2(NULL, 23.4, NULL) from dual;

NVL2(NULL,23.4,NULL)
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _


SQL> select NVL2(56.2, 23.4, NULL) from dual;

NVL2(56.2,23.4,NULL)
_ _ _ _ _ _ _ _ _ _ _ _ _ _
             23.4

SQL > select * from sal;
ID   NAME                 SAL       COMMISION_PERCENTAGE
_ _   _ _ _ _ _ _ _ _      _ _ _      _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
 1    Joe Root            75000            0.15
 2    Ros Taylor           90000            0.1
 3    Paul Adams            50000            0.01
 4    Victor Lee           43000
 5    Matt Potts           20000
 6    James Anderson        200000           0.33
 7    Craig Overton         11000
 8    Rory Burns            9000
8 rows selected.
SQL> SELECT  id, name, sal, NVL2(COMMISION_PERCENTAGE, sal, 0),
  2           (sal) + (sal * NVL2(COMMISION_PERCENTAGE, NULL, 0.1))
  3            Wierd_salary FROM sal;

ID   NAME                 SAL     NVL2(COMMISION_PERCENTAGE,sal,0)   WEIRD_SALARY
        _ _   _ _ _ _ _ _ _ _      _ _ _    _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
       1    Joe Root            75000           75000
       2    Ros Taylor           90000           90000
       3    Paul Adams            50000            50000
       4    Victor Lee          43000            0                        47300
```

```
5   Matt Potts          20000          0                      22000
6   James Anderson        200000           200000
7   Craig Overton        11000           0                      12100

8   Rory Burns           9000           0                      9900
```

3.) DECODE()

This is also one of the expressions used in SQL. This Decode expression is used as IF, ELSE IF, ELSE IF Ladder style. This decode works on the basis of the condition specified.

Any kind of operation specified is going to work here.

The input types must chosen based on the data types specified.

Syntax

```
DECODE (column  name,  number 1 to be searched, result 1 to be updated
                  , number 2 to be searched, result 2 to be updated
                   , number 3 to be searched, result 3 to be updated
              . . . . . . . . .
                   number n to be searched, result n to be updated , default)
```

Example Queries

```
SQL > select * from ipla;


SID  SNAME           SAL       AGE
_ _ _    _ _ _ _ _      _ _ _ _    _ _ _ _
     1   mahi            12          40
     2  kohli            14           33
     3   DK              6.25        33
     4  warner         6.75        33
     5  rahul            16           29
     6  pandya         14          27

SQL > SELECT Sname, sid, sal,
  2        DECODE (sid, 1, 1.5*sal,
  3                            2, 4*sal,
  4                         3, 9*sal,
  5                         4, 10.25*sal,
  6              sal)
  7         "REVISED SALARY"
  8  from ipla ;

SNAME        SID       SAL        REVISED SALARY
___     ______      ____    ___________
mahi              1        12            18
kohli             2        14            56
DK                3      6.25           56.25
warner          4      6.75           69.1875
rahul             5        16            16
pandya            6        14            14

6 rows selected.
SQL > select * from ipla;


SID  SNAME             SAL        AGE
___     ______      ____    ____
   1   mahi              12          40
   2  kohli              14           33
   3   DK                6.25        33
   4  warner          6.75        33
   5  rahul             16           29
   6  pandya          14           27
   7      Tim David      8.25        26
SQL > SELECT Sid, Sname, Sal, Age,
  2             DECODE (sid, 1, 2 * sid * sal,
  3                         2, 3 * sid * sal,
  4                         3, 5 * sid * sal,
  5                         4, 10 * sid * sal,
  6                         5, 12 * sid * sal,
  7                         6, 15 * sid * sal,
  8                        sal/2)
  9             "UPGRADED SALARY"
```

```
10      from ipla ;

SID  SNAME            SAL       AGE      UPGRADED SALARY
_ _ _    _ _ _ _ _       _ _ _ _    _ _ _ _    _ _ _ _ _ _ _ _ _ _ _ _
    1   mahi             12           40            24
    2  kohli             14           33            84
    3  DK               6.25          33         93.75
    4   warner        6.75          33          270
      5     rahul            16           29            960
      6     pandya            14          27           1260
      7     Tim David      8.25        26          4.125

7 rows selected.
```

4.) COALESCE()

This also one of the expression used in SQL. This expression works similar to NVL () expression. The only difference it can accept inputs greater than two. It returns the first NOT NULL input element.

The input data type can be anything. The inputs can be int, float, string, character, number, etc.

Syntax

```
COALESCE (input 1, input 2, input 3, . . . . . . . . . ., input n)
```

Example Queries

```
SQL> select COALESCE(NULL, 1) from dual;

COALESCE(NULL,1)
_ _ _ _ _ _ _ _ _ _ _ _
           1
SQL> select COALESCE(1, 2, 2) from dual;

COALESCE(1,2,2)
_ _ _ _ _ _ _ _ _ _ _ _
          1

```

```
SQL> select COALESCE(NULL, 2, 2) from dual;

COALESCE(NULL,2,2)
_ _ _ _ _ _ _ _ _ _ _ _ _
            2

SQL> select COALESCE(NULL, NULL, 2) from dual;

COALESCE(NULL,NULL,2)
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
              2

SQL> select COALESCE(NULL, NULL, NULL) from dual;

C
_


SQL> select COALESCE(NULL, NULL, NULL, 1, 2, 3, 4, 5, 6) from dual;

COALESCE(NULL,NULL,NULL,1,2,3,4,5,6)
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
                          1
SQL> select COALESCE(NULL, 'NULL', 'Stuart Broad', 'Adam Gilchrist') from dual;


COAL
_ _ _ _
NULL
```

5.) LNNVL()

This is one of the function of SQL which is used in SQL. This is used to convert True to False and False to True.

The LNNVL () function has the capacity to hold a condition. This makes the condition go reverse.

If the condition is SID = 2. Then LNNVL (SID = 2) is equivalent to SID ! = 2.

Syntax

```
LNNVL (Condition)
```

Example Queries

```
SQL> select * from ipla;

SID  SNAME              SAL        AGE
___    _ _ _ _ _ _     _ _ _ _    _ _ _ _
 1   mahi               12         40
 2  kohli              14         33
 3   DK                6.25       33
 4  warner              6.75       33
 5  rahul              16         29
 6  pandya              14         27
 7  Tim David           8.25       26

7 rows selected.

SQL> select * from ipla where sid=2;
SID  SNAME             SAL        AGE
___    _ _ _ _ _ _      _ _ _ _    _ _ _ _
  2      kohli            14            33

SQL> select * from ipla where LNNVL (sid = 2) ;

SID  SNAME             SAL        AGE
_ _ _ _ _ _ _ _ _      _ _ _ _    _ _ _ _
1    mahi           12          40
3    DK            6.25        33
4   warner          6.75        33
5   rahul          16          29
6   pandya          14          27
7   Tim David      8.25         26
6 rows selected.
```

```
SQL> select * from sal;
ID   NAME                 SAL       COMMISION_PERCENTAGE
__  _________        ___        _____________________
 1    Joe Root            75000            0.15
 2    Ros Taylor         90000              0.1
 3    Paul Adams        50000            0.01
 4    Victor Lee         43000
 5    Matt Potts           20000
 6    James Anderson    200000           0.33
7    Craig Overton       11000
 8    Rory Burns          9000
8 rows selected


SQL> select * from sal where NVL (COMMISION_PERCENTAGE, 0) =0;
ID   NAME                 SAL       COMMISION_PERCENTAGE
__  _________        ___        _____________________
4    Victor Lee          43000
5    Matt Potts           20000
7    Craig Overton          11000
 8    Rory Burns          9000
4 rows selected.
SQL> select * from sal where LNNVL (NVL (COMMISION_PERCENTAGE, 0)  = 0) ;

ID   NAME                 SAL       COMMISION_PERCENTAGE
__  _________        ___        _____________________
 1    Joe Root            75000            0.15
 2    Ros Taylor             90000             0.1
 3    Paul Adams              50000             0.01
6    James Anderson          200000           0.33
4 rows selected
```

6.) NANVL ()

If the input value n2 is NaN (not a number), this method returns an alternative value n1, and if n2 is not NaN, it returns n2. Only floating-point numbers of the types BINARY FLOAT or BINARY DOUBLE can be used with this function.

The function accepts any numeric or nonnumeric data type as an input, with the ability to implicitly convert to a numeric data type.

The method returns BINARY DOUBLE if the parameter is BINARY FLOAT. If not, the function returns a numeric data type that matches the parameter.

Syntax

```
NANVL (input 1, input 2)
```

Example Queries

```
SQL> SELECT * FROM FPT;

   DEC_NUM  BIN_DOUBLE          BIN_FLOAT
_ _ _ _ _ _ _    _ _ _ _ _ _ _ _ _     _ _ _ _ _ _ _
 3563.971   3.564E+003        3.564E+003
SQL> INSERT INTO FPT VALUES (0, 'NaN', 'NaN');

1 row created.

SQL> SELECT * FROM FPT ;

   DEC_NUM  BIN_DOUBLE          BIN_FLOAT
_ _ _ _ _ _ _    _ _ _ _ _ _ _ _ _     _ _ _ _ _ _ _
 3563.971   3.564E+003        3.564E+003
      0         Nan             Nan
SQL> SELECT bin_float, NANVL (bin_float,0)
2         FROM float_point_test;

BIN_FLOAT    NANVL(BIN_FLOAT,0)
_ _ _ _ _ _ _     _ _ _ _ _ _ _ _ _ _ _ _ _ _
1.514E+003        1.514E+003
      Nan                  0


MULTI QUERY APPROACH
SQL> select * from ipla;

```

```

SID  SNAME          SAL       AGE
___   ______      ____    ____
     1   mahi          12          40
     2  kohli         14          33
     3   DK           6.25        33
     4  warner         6.75        33
     5  rahul         16          29
     6  pandya         14          27
     7   Tim David      8.25        26
SQL> select Sname, sid, sal,
  2         DECODE (SID, 1,  (sal + sid * 5),
  3                        2, (sal + sid*4),
  4                        3, NVL(sal*4, NULL),
  5                        4, NVL2(NULL, sal*3, sal*4),
  6                        5, COALESCE(NULL, NULL, sal+4),
  7                        6, 3 * sal,
  8                        sal*2)
  9                        "REVISED SALARY "
 10                       from ipla;


SID  SNAME          SAL       REVISED SALARY
_________      ____     ___________
     1   mahi          12           17
     2  kohli         14          22
     3   DK           6.25         25
     4  warner         6.75         27
     5  rahul         16          20
     6  pandya         14           42
     7   Tim David      8.25          16.5
```

This is all about SQL general functions: NVL, NVL2, DECODE, COALESCE, NULLIF, LNNVL and NANVL