NUMERIC FUNCTIONS IN SQL

Shadab Khan

Contents

ABS(X)

The ABS() function returns the absolute value of X. Consider the following example –

```
SQL> SELECT ABS(2);

+---------------------------------------------------------+

| ABS(2)                                                  |

+---------------------------------------------------------+

| 2                                                       |

+---------------------------------------------------------+

1 row in set (0.00 sec)


SQL> SELECT ABS(-2);

+---------------------------------------------------------+

| ABS(2)                                                  |

+---------------------------------------------------------+

| 2                                                       |

+---------------------------------------------------------+

1 row in set (0.00 sec)
```

ACOS(X)

This function returns the arccosine of X. The value of X must range between -1 and 1 or NULL will be returned. Consider the following example –

```
SQL> SELECT ACOS(1);

+---------------------------------------------------------+

| ACOS(1)                                                 |

+---------------------------------------------------------+

| 0.000000                                                |

+---------------------------------------------------------+

1 row in set (0.00 sec)
```

ASIN(X)

The ASIN() function returns the arcsine of X. The value of X must be in the range of -1 to 1 or NULL is returned.

```
SQL> SELECT ASIN(1);
+---------------------------------------------------------+
| ASIN(1)                                                 |
+---------------------------------------------------------+
| 1.5707963267949                                         |
+---------------------------------------------------------+
1 row in set (0.00 sec)
```

ATAN(X)

This function returns the arctangent of X.

```
SQL> SELECT ATAN(1);
+---------------------------------------------------------+
| ATAN(1)                                                 |
+---------------------------------------------------------+
| 0.78539816339745                                        |
+---------------------------------------------------------+
1 row in set (0.00 sec)
```

ATAN2(Y,X)

This function returns the arctangent of the two arguments: X and Y. It is similar to the arctangent of Y/X, except that the signs of both are used to find the quadrant of the result.

```
SQL> SELECT ATAN2(3,6);
+---------------------------------------------------------+
| ATAN2(3,6)                                              |
+---------------------------------------------------------+
| 0.46364760900081                                        |
+---------------------------------------------------------+
1 row in set (0.00 sec)
```

BIT_AND(expression)

The BIT_AND function returns the bitwise AND of all bits in expression. The basic premise is that if two corresponding bits are the same, then a bitwise AND operation will return 1, while if they are different, a bitwise AND operation will return 0. The function itself returns a 64-bit integer value. If there are no matches, then it will return 18446744073709551615. The following example performs the BIT_AND function on the PRICE column grouped by the MAKER of the car –

```
SQL> SELECT
          MAKER, BIT_AND(PRICE) BITS
          FROM CARS GROUP BY MAKER
+-----------------------------------------------------------+
|MAKER           BITS                                       |
+-----------------------------------------------------------+
|CHRYSLER        512                                        |
|FORD            12488                                      |
|HONDA           2144                                       |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

BIT_COUNT(numeric_value)

The BIT_COUNT() function returns the number of bits that are active in numeric_value. The following example demonstrates using the BIT_COUNT() function to return the number of active bits for a range of numbers –

```
SQL> SELECT
          BIT_COUNT(2) AS TWO,
          BIT_COUNT(4) AS FOUR,
          BIT_COUNT(7) AS SEVEN
+-----+------+-------+
| TWO | FOUR | SEVEN |
+-----+------+-------+
|   1 |    1 |     3 |
+-----+-----+-------+
1 row in set (0.00 sec)
```

BIT_OR(expression)

The BIT_OR() function returns the bitwise OR of all the bits in expression. The basic premise of the bitwise OR function is that it returns 0 if the corresponding bits match and 1 if they do not. The function returns a 64-bit integer, and if there are no matching rows, then it returns 0. The following example performs the BIT_OR() function on the PRICE column of the CARS table, grouped by the MAKER –

```
SQL> SELECT
          MAKER, BIT_OR(PRICE) BITS
          FROM CARS GROUP BY MAKER
+----------------------------------------------------------+
|MAKER           BITS                                      |
+----------------------------------------------------------+
|CHRYSLER        62293                                     |
|FORD            16127                                     |
|HONDA           32766                                     |
+----------------------------------------------------------+
1 row in set (0.00 sec)
```

CEIL(X)
CEILING(X)

These functions return the smallest integer value that is not smaller than X. Consider the following example –

```
SQL> SELECT CEILING(3.46);
+----------------------------------------------------------+
| CEILING(3.46)                                            |
+----------------------------------------------------------+
| 4                                                        |
+----------------------------------------------------------+
1 row in set (0.00 sec)


SQL> SELECT CEIL(-6.43);
+---------------------------------------------------------+
| CEIL(-6.43)                                             |
+---------------------------------------------------------+
| -6                                                      |
+---------------------------------------------------------+
1 row in set (0.00 sec)
```

CONV(N,from_base,to_base)

The purpose of the CONV() function is to convert numbers between different number bases. The function returns a string of the value N converted from from_base to to_base. The minimum base value is 2 and the maximum is 36. If any of the arguments are NULL, then the function returns NULL. Consider the following example, which converts the number 5 from base 16 to base 2 –

```
SQL> SELECT CONV(5,16,2);
+---------------------------------------------------------+
| CONV(5,16,2)                                            |
+---------------------------------------------------------+
| 101                                                     |
+---------------------------------------------------------+
1 row in set (0.00 sec)
```

COS(X)

This function returns the cosine of X. The value of X is given in radians.

```
SQL>SELECT COS(90);
+---------------------------------------------------------+
| COS(90)                                                 |
+---------------------------------------------------------+
| -0.44807361612917                                       |
+----------------------------------------------------------+
1 row in set (0.00 sec)
```

COT(X)

This function returns the cotangent of X. Consider the following example −

```
SQL>SELECT COT(1);
+----------------------------------------------------------+
| COT(1)                                                   |
+----------------------------------------------------------+
| 0.64209261593433                                         |
+----------------------------------------------------------+
1 row in set (0.00 sec)
```

DEGREES(X)

This function returns the value of X converted from radians to degrees.

```
SQL>SELECT DEGREES(PI());
+----------------------------------------------------------+
| DEGREES(PI())                                            |
+----------------------------------------------------------+
| 180.000000                                               |
+----------------------------------------------------------+
1 row in set (0.00 sec)
```

EXP(X)

This function returns the value of e (the base of the natural logarithm) raised to the power of X.

```
SQL>SELECT EXP(3);
+----------------------------------------------------------+
| EXP(3)                                                   |
+----------------------------------------------------------+
| 20.085537                                                |
+----------------------------------------------------------+
1 row in set (0.00 sec)
```

FLOOR(X)

This function returns the largest integer value that is not greater than X.

```
SQL>SELECT FLOOR(7.55);

+---------------------------------------------------------+

| FLOOR(7.55)                                             |

+---------------------------------------------------------+

| 7                                                       |

+---------------------------------------------------------+

1 row in set (0.00 sec)
```

FORMAT(X,D)

The FORMAT() function is used to format the number X in the following format: ###,###,###.## truncated to D decimal places. The following example demonstrates the use and output of the FORMAT() function –

```
SQL>SELECT FORMAT(423423234.65434453,2);

+---------------------------------------------------------+

| FORMAT(423423234.65434453,2)                            |

+---------------------------------------------------------+

| 423,423,234.65                                          |

+---------------------------------------------------------+

1 row in set (0.00 sec)
```

GREATEST(n1,n2,n3,..........)

The GREATEST() function returns the greatest value in the set of input parameters (n1, n2, n3, a nd so on). The following example uses the GREATEST() function to return the largest number from a set of numeric values –

```
SQL>SELECT GREATEST(3,5,1,8,33,99,34,55,67,43);

+---------------------------------------------------------+

| GREATEST(3,5,1,8,33,99,34,55,67,43)                     |
+---------------------------------------------------------+
| 99                                                      |
+---------------------------------------------------------+
1 row in set (0.00 sec)
```

INTERVAL(N,N1,N2,N3,..........)

The INTERVAL() function compares the value of N to the value list (N1, N2, N3, and so on). The function returns 0 if N < N1, 1 if N < N2, 2 if N <N3, and so on. It will return .1 if N is NULL. The value list must be in the form N1 < N2 < N3 in order to work properly. The following code is a simple example of how the INTERVAL() function works −

```
SQL>SELECT INTERVAL(6,1,2,3,4,5,6,7,8,9,10);
+---------------------------------------------------------+
| INTERVAL(6,1,2,3,4,5,6,7,8,9,10)                        |
+---------------------------------------------------------+
| 6                                                       |
+---------------------------------------------------------+
1 row in set (0.00 sec)
```

Remember that 6 is the zero-based index in the value list of the first value that was greater than N. In our case, 7 was the offending value and is located in the sixth index slot.

LEAST(N1,N2,N3,N4,......)

The LEAST() function is the opposite of the GREATEST() function. Its purpose is to return the least-valued item from the value list (N1, N2, N3, and so on). The following example shows the proper usage and output for the LEAST() function −

```
SQL>SELECT LEAST(3,5,1,8,33,99,34,55,67,43);
+---------------------------------------------------------+
| LEAST(3,5,1,8,33,99,34,55,67,43)                        |
+---------------------------------------------------------+
| 1                                                       |
+---------------------------------------------------------+
1 row in set (0.00 sec)
```

LOG(X)
LOG(B,X)

The single argument version of the function will return the natural logarithm of X. If it is called with two arguments, it returns the logarithm of X for an arbitrary base B. Consider the following example –

```
SQL>SELECT LOG(45);
+---------------------------------------------------------+
| LOG(45)                                                 |
+---------------------------------------------------------+
| 3.806662                                                |
+---------------------------------------------------------+
1 row in set (0.00 sec)

SQL>SELECT LOG(2,65536);
+---------------------------------------------------------+
| LOG(2,65536)                                            |
+---------------------------------------------------------+
| 16.000000                                               |
+---------------------------------------------------------+
1 row in set (0.00 sec)
```

LOG10(X)

This function returns the base-10 logarithm of X.

```
SQL>SELECT LOG10(100);
+---------------------------------------------------------+
| LOG10(100)                                              |
+---------------------------------------------------------+
| 2.000000                                                |
+---------------------------------------------------------+
1 row in set (0.00 sec)
```

MOD(N,M)

This function returns the remainder of N divided by M. Consider the following example –

```
SQL>SELECT MOD(29,3);
+---------------------------------------------------------+
| MOD(29,3)                                               |
+---------------------------------------------------------+
| 2                                                       |
+---------------------------------------------------------+
1 row in set (0.00 sec)
```

OCT(N)

The OCT() function returns the string representation of the octal number N. This is equivalent to using CONV(N,10,8).

```
SQL>SELECT OCT(12);
+---------------------------------------------------------+
| OCT(12)                                                 |
+---------------------------------------------------------+
| 14                                                      |
+---------------------------------------------------------+
1 row in set (0.00 sec)
```

PI()

This function simply returns the value of pi. SQL internally stores the full double-precision value of pi.

```
SQL>SELECT PI();
+---------------------------------------------------------+
| PI()                                                    |
+---------------------------------------------------------+
| 3.141593                                                |
+---------------------------------------------------------+

1 row in set (0.00 sec)
```

POW(X,Y)

POWER(X,Y)

These two functions return the value of X raised to the power of Y.

```
SQL> SELECT POWER(3,3);

+---------------------------------------------------------+

| POWER(3,3)                                              |

+---------------------------------------------------------+

| 27                                                      |

+---------------------------------------------------------+

1 row in set (0.00 sec)
```

RADIANS(X)

This function returns the value of X, converted from degrees to radians.

```
SQL>SELECT RADIANS(90);

+---------------------------------------------------------+

| RADIANS(90)                                             |

+---------------------------------------------------------+

|1.570796                                                 |

+---------------------------------------------------------+

1 row in set (0.00 sec)
```

ROUND(X)

ROUND(X,D)

This function returns X rounded to the nearest integer. If a second argument, D, is supplied, then the function returns X rounded to D decimal places. D must be positive or all digits to the right of the decimal point will be removed. Consider the following example –

```
SQL>SELECT ROUND(5.693893);
```

```
+---------------------------------------------------------+
| ROUND(5.693893)                                         |
+---------------------------------------------------------+
| 6                                                       |
+---------------------------------------------------------+
1 row in set (0.00 sec)

SQL>SELECT ROUND(5.693893,2);
+---------------------------------------------------------+
| ROUND(5.693893,2)                                       |
+---------------------------------------------------------+
| 5.69                                                    |
+---------------------------------------------------------+
1 row in set (0.00 sec)
```

SIGN(X)

This function returns the sign of X (negative, zero, or positive) as -1, 0, or 1.

```
SQL>SELECT SIGN(-4.65);
+---------------------------------------------------------+
| SIGN(-4.65)                                             |
+---------------------------------------------------------+
| -1                                                      |
+---------------------------------------------------------+
1 row in set (0.00 sec)

SQL>SELECT SIGN(0);
+---------------------------------------------------------+
| SIGN(0)                                                 |
+---------------------------------------------------------+
| 0                                                       |
+---------------------------------------------------------+
1 row in set (0.00 sec)
```

```
SQL>SELECT SIGN(4.65);
+-----------------------------------------------------------+
| SIGN(4.65)                                                |
+-----------------------------------------------------------+
| 1                                                         |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

SIN(X)

This function returns the sine of X. Consider the following example –

```
SQL>SELECT SIN(90);
+-----------------------------------------------------------+
| SIN(90)                                                   |
+-----------------------------------------------------------+
| 0.893997                                                  |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

SQRT(X)

This function returns the non-negative square root of X. Consider the following example –

```
SQL>SELECT SQRT(49);
+-----------------------------------------------------------+
| SQRT(49)                                                  |
+-----------------------------------------------------------+
| 7                                                         |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

STD(expression)

STDDEV(expression)

The STD() function is used to return the standard deviation of expression. This is equivalent to taking the square root of the VARIANCE() of expression. The following example computes the standard deviation of the PRICE column in our CARS table –

```
SQL>SELECT STD(PRICE) STD_DEVIATION FROM CARS;

+---------------------------------------------------------+

| STD_DEVIATION                                           |

+---------------------------------------------------------+

| 7650.2146                                               |

+---------------------------------------------------------+

1 row in set (0.00 sec)
```

TAN(X)

This function returns the tangent of the argument X, which is expressed in radians.

```
SQL>SELECT TAN(45);

+---------------------------------------------------------+

| TAN(45)                                                 |

+---------------------------------------------------------+

| 1.619775                                                |

+---------------------------------------------------------+

1 row in set (0.00 sec)
```

TRUNCATE(X,D)

This function is used to return the value of X truncated to D number of decimal places. If D is 0, then the decimal point is removed. If D is negative, then D number of values in the integer part of the value is truncated. Consider the following example –

```
SQL>SELECT TRUNCATE(7.536432,2);

+---------------------------------------------------------+

| TRUNCATE(7.536432,2)                                    |

+---------------------------------------------------------+
| 7.53                                                       |
+------------------------------------------------------------+
1 row in set (0.00 sec)
```