Oracle COALESCE Function

The Oracle/PLSQL COALESCE function returns the first non-null expression in the list. If all expressions evaluate to null, then the COALESCE function will return null.

It requires at least two expressions. In case all expressions evaluate to null, the function returns null.

The COALESCE function returns any datatype such as a string, numeric, date, etc. (BUT all expressions must be the same datatype in the COALESCE function.) If all expressions are not the same datatype, an ORA-00932 error will be returned.

Syntax

The syntax for the COALESCE function in Oracle/PLSQL is:

```
COALESCE( expr1, expr2, ... expr_n )
```

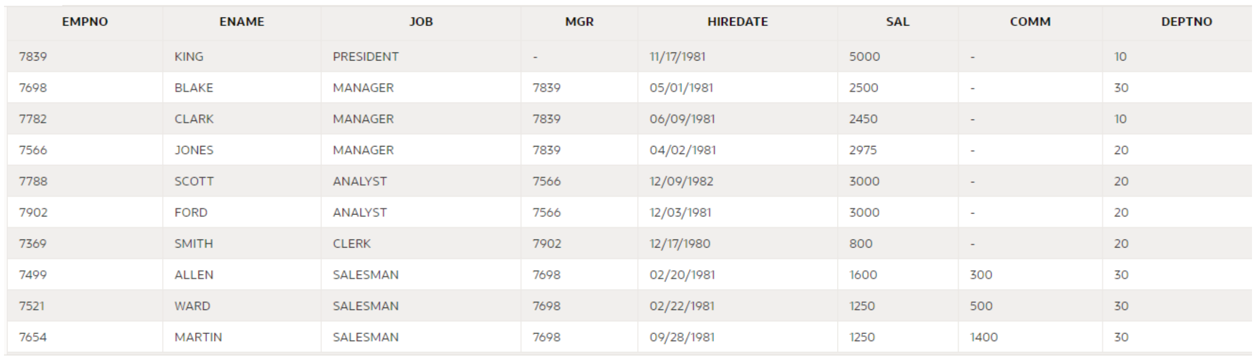
The expressions to test for non-null values. The expressions must all be the same datatype.

The COALESCE function can be used in the following versions of Oracle/PLSQL:
Oracle 12c, Oracle 11g, Oracle 10g, Oracle 9i

Example

The "emp" table is taking for the example of this function.

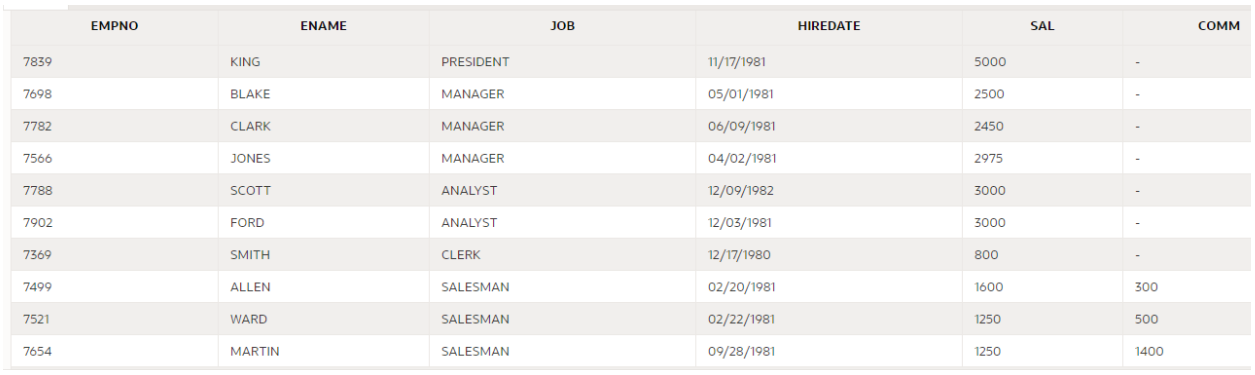
```
SELECT * FROM emp;
```



EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7839	KING	PRESIDENT	-	11/17/1981	5000	-	10
7698	BLAKE	MANAGER	7839	05/01/1981	2500	-	30
7782	CLARK	MANAGER	7839	06/09/1981	2450	-	10
7566	JONES	MANAGER	7839	04/02/1981	2975	-	20
7788	SCOTT	ANALYST	7566	12/09/1982	3000	-	20
7902	FORD	ANALYST	7566	12/03/1981	3000	-	20
7369	SMITH	CLERK	7902	12/17/1980	800	-	20
7499	ALLEN	SALESMAN	7698	02/20/1981	1600	300	30
7521	WARD	SALESMAN	7698	02/22/1981	1250	500	30
7654	MARTIN	SALESMAN	7698	09/28/1981	1250	1400	30

The selected data items of the table are taken for the simplicity.

```
SELECT empno, ename, job, hiredate, sal, comm FROM emp;
```



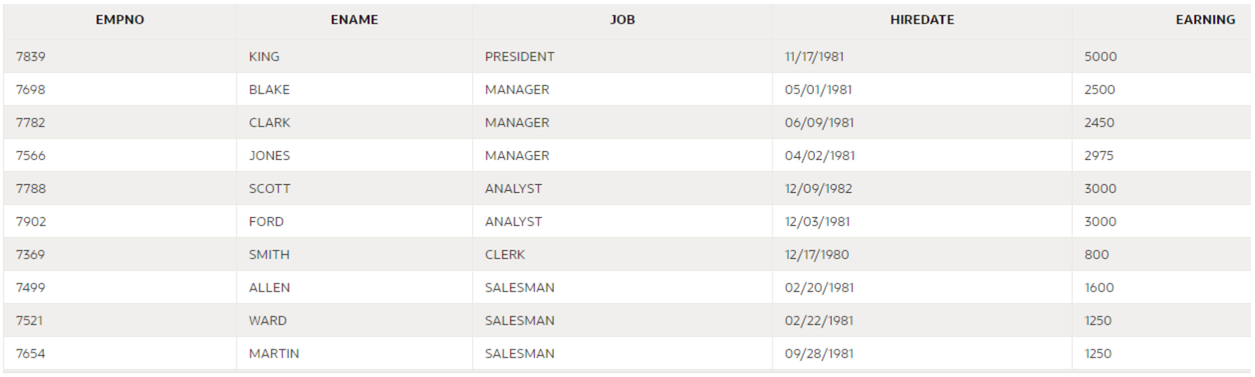
EMPNO	ENAME	JOB	HIREDATE	SAL	COMM
7839	KING	PRESIDENT	11/17/1981	5000	-
7698	BLAKE	MANAGER	05/01/1981	2500	-
7782	CLARK	MANAGER	06/09/1981	2450	-
7566	JONES	MANAGER	04/02/1981	2975	-
7788	SCOTT	ANALYST	12/09/1982	3000	-
7902	FORD	ANALYST	12/03/1981	3000	-
7369	SMITH	CLERK	12/17/1980	800	-
7499	ALLEN	SALESMAN	02/20/1981	1600	300
7521	WARD	SALESMAN	02/22/1981	1250	500
7654	MARTIN	SALESMAN	09/28/1981	1250	1400

COALESCE function is applying in the query. Both “sal” and “comm” have same data types.

```
SELECT empno, ename, job, hiredate,

COALESCE(sal, comm) EARNING FROM emp;
```

Let me explain this output. First, the function checks in the “sal” column, it finds the null value then goes to the other column “comm” for checking. If the value exists then it displays. Otherwise, it places the null value.
But, in this case, there is no null value in the column “sal” so, there is no change in the values of the “sal” column.



EMPNO	ENAME	JOB	HIREDATE	EARNING
7839	KING	PRESIDENT	11/17/1981	5000
7698	BLAKE	MANAGER	05/01/1981	2500
7782	CLARK	MANAGER	06/09/1981	2450
7566	JONES	MANAGER	04/02/1981	2975
7788	SCOTT	ANALYST	12/09/1982	3000
7902	FORD	ANALYST	12/03/1981	3000
7369	SMITH	CLERK	12/17/1980	800
7499	ALLEN	SALESMAN	02/20/1981	1600
7521	WARD	SALESMAN	02/22/1981	1250
7654	MARTIN	SALESMAN	09/28/1981	1250

Now, two rows updated with null values.

```
UPDATE emp SET sal = null WHERE empno = ‘7698’;

UPDATE emp SET sal = null WHERE empno = ‘7499’;
```

After updating the rows, both rows (2nd and 8th) have null values.

EMPNO	ENAME	JOB	HIREDATE	SAL	COMM
7839	KING	PRESIDENT	11/17/1981	5000	-
7698	BLAKE	MANAGER	05/01/1981	-	-
7782	CLARK	MANAGER	06/09/1981	2450	-
7566	JONES	MANAGER	04/02/1981	2975	-
7788	SCOTT	ANALYST	12/09/1982	3000	-
7902	FORD	ANALYST	12/03/1981	3000	-
7369	SMITH	CLERK	12/17/1980	800	-
7499	ALLEN	SALESMAN	02/20/1981	-	300
7521	WARD	SALESMAN	02/22/1981	1250	500
7654	MARTIN	SALESMAN	09/28/1981	1250	1400

The same query with the COALESCE function executes, get the output checked.

```
SELECT empno, ename, job, hiredate,
COALESCE(sal, comm) EARNING FROM emp;
```

The 2nd row has a null value, the reason, there is no value in “sal” and “comm” whereas, the query found null value in “sal” of the 8th row, then “comm” value is replaced, that is 300.

It is a very important point. The similar data types are needed for all data items under the COALESCE function.

EMPNO	ENAME	JOB	HIREDATE	EARNING
7839	KING	PRESIDENT	11/17/1981	5000
7698	BLAKE	MANAGER	05/01/1981	-
7782	CLARK	MANAGER	06/09/1981	2450
7566	JONES	MANAGER	04/02/1981	2975
7788	SCOTT	ANALYST	12/09/1982	3000
7902	FORD	ANALYST	12/03/1981	3000
7369	SMITH	CLERK	12/17/1980	800
7499	ALLEN	SALESMAN	02/20/1981	300
7521	WARD	SALESMAN	02/22/1981	1250
7654	MARTIN	SALESMAN	09/28/1981	1250

The above COALESCE function is equivalent to the following IF-THEN-ELSE statement:

```
IF sal is not null THEN
```

```
   result := sal;


ELSIF comm is not null THEN

   result := comm;

ELSE

   result := null;

END IF;
```

The COALESCE function compares each value, one by one.

Short-circuit evaluation

The COALESCE() function uses short-circuit evaluation. It means that the function stops evaluating the remaining expressions once it finds the first one evaluates to a non-null value. Consider the following example:

```
SELECT COALESCE(1+2, 1/0) FROM dual;
```

In this example, the COALESCE() function only evaluated the first expression because the result of the first expression was three (1+2). It did not evaluate the second expression (1/0). If it had done so, Oracle would have issued the division by zero error.

Oracle COALESCE() and CASE expression

You can use the COALESCE() function instead of the longer CASE expression when it comes to test for null in multiple expressions. The COALESCE() function is more concise than a CASE expression that involves null evaluations.

If you check for NULL in two expressions, the COALESCE() function is equivalent to the CASE expression.

For example, the following COALESCE() function:

```
COALESCE(e1,e2);

is equivalent to:
 CASE
        WHEN e1 IS NOT NULL THEN e1
        ELSE e2
        END

Likewise,
COALESCE(e1,e2,. . .,en);
is equivalent to:

  CASE
        WHEN e1 IS NOT NULL THEN e1
        ELSE
        COALESCE(e2, . . .,en)
        END
```

Oracle COALESCE() vs. NVL()

The COALESCE() function is a part of SQL ANSI-92 standard while NVL() function is Oracle specific.

In case of two expressions, the COALESCE() function and NVL() seems to be similar but their implementations are different. See the following statements:

```
SELECT COALESCE(1, NULL) FROM dual;
```

```
SELECT NVL(1, NULL) FROM dual;
```

Both statements return the same result which is one. However, the COALESCE() function only evaluates the first expression to determine the result while the NVL() function evaluates both expressions.
Let’s see the following example:

```
SELECT COALESCE(1, 1/0) FROM dual;
```

The statement above returned 1 whereas the following example causes an error:

```
SELECT NVL(1, 1/0) FROM dual;
```

The error is:
"ORA-01476: divisor is equal to zero"

Because the NVL() function evaluated the second expression 1/0 that causes the error.
In this tutorial, you have learned how to use the Oracle COALESCE() function to return the first non-null expression in a list of expressions.