

SQL - UNIONS CLAUSE

<http://www.tutorialspoint.com/sql/sql-unions-clause.htm>

Copyright © tutorialspoint.com

The SQL **UNION** clause/operator is used to combine the results of two or more SELECT statements without returning any duplicate rows.

To use UNION, each SELECT must have the same number of columns selected, the same number of column expressions, the same data type, and have them in the same order, but they do not have to be the same length.

Syntax:

The basic syntax of **UNION** is as follows:

```
SELECT column1 [, column2 ]
FROM table1 [, table2 ]
[WHERE condition]

UNION

SELECT column1 [, column2 ]
FROM table1 [, table2 ]
[WHERE condition]
```

Here given condition could be any given expression based on your requirement.

Example:

Consider the following two tables, a CUSTOMERS table is as follows:

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00

b Another table is ORDERS as follows:

OID	DATE	CUSTOMER_ID	AMOUNT
102	2009-10-08 00:00:00	3	3000
100	2009-10-08 00:00:00	3	1500
101	2009-11-20 00:00:00	2	1560
103	2008-05-20 00:00:00	4	2060

Now, let us join these two tables in our SELECT statement as follows:

```
SQL> SELECT ID, NAME, AMOUNT, DATE
FROM CUSTOMERS
LEFT JOIN ORDERS
ON CUSTOMERS.ID = ORDERS.CUSTOMER_ID
UNION
SELECT ID, NAME, AMOUNT, DATE
FROM CUSTOMERS
RIGHT JOIN ORDERS
```

```
ON CUSTOMERS.ID = ORDERS.CUSTOMER_ID;
```

This would produce the following result:

ID	NAME	AMOUNT	DATE
1	Ramesh	NULL	NULL
2	Khilan	1560	2009-11-20 00:00:00
3	kaushik	3000	2009-10-08 00:00:00
3	kaushik	1500	2009-10-08 00:00:00
4	Chaitali	2060	2008-05-20 00:00:00
5	Hardik	NULL	NULL
6	Komal	NULL	NULL
7	Muffy	NULL	NULL

The UNION ALL Clause:

The UNION ALL operator is used to combine the results of two SELECT statements including duplicate rows.

The same rules that apply to UNION apply to the UNION ALL operator.

Syntax:

The basic syntax of **UNION ALL** is as follows:

```
SELECT column1 [, column2 ]  
FROM table1 [, table2 ]  
[WHERE condition]
```

```
UNION ALL
```

```
SELECT column1 [, column2 ]  
FROM table1 [, table2 ]  
[WHERE condition]
```

Here given condition could be any given expression based on your requirement.

Example:

Consider the following two tables, *a* CUSTOMERS table is as follows:

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00

b Another table is ORDERS as follows:

OID	DATE	CUSTOMER_ID	AMOUNT
102	2009-10-08 00:00:00	3	3000
100	2009-10-08 00:00:00	3	1500
101	2009-11-20 00:00:00	2	1560
103	2008-05-20 00:00:00	4	2060

Now, let us join these two tables in our SELECT statement as follows:

```
SQL> SELECT ID, NAME, AMOUNT, DATE
      FROM CUSTOMERS
      LEFT JOIN ORDERS
      ON CUSTOMERS.ID = ORDERS.CUSTOMER_ID
UNION ALL
      SELECT ID, NAME, AMOUNT, DATE
      FROM CUSTOMERS
      RIGHT JOIN ORDERS
      ON CUSTOMERS.ID = ORDERS.CUSTOMER_ID;
```

This would produce the following result:

ID	NAME	AMOUNT	DATE
1	Ramesh	NULL	NULL
2	Khilan	1560	2009-11-20 00:00:00
3	kaushik	3000	2009-10-08 00:00:00
3	kaushik	1500	2009-10-08 00:00:00
4	Chaitali	2060	2008-05-20 00:00:00
5	Hardik	NULL	NULL
6	Komal	NULL	NULL
7	Muffy	NULL	NULL
3	kaushik	3000	2009-10-08 00:00:00
3	kaushik	1500	2009-10-08 00:00:00
2	Khilan	1560	2009-11-20 00:00:00
4	Chaitali	2060	2008-05-20 00:00:00

There are two other clauses *i. e.*, *operators*, which are very similar to UNION clause:

- SQL [INTERSECT Clause](#): is used to combine two SELECT statements, but returns rows only from the first SELECT statement that are identical to a row in the second SELECT statement.
- SQL [EXCEPT Clause](#) : combines two SELECT statements and returns rows from the first SELECT statement that are not returned by the second SELECT statement.

Loading [Mathjax]/jax/output/HTML-CSS/jax.js