# **The SQL UNION Operator**

**The SQL UNION Operator:**

1. The UNION operator is used to combine the result-set of two or more SELECT statements.  
   -Every SELECT statement within UNION must have the same number of columns.  
   -The columns must also have similar data types.  
   -The columns in every SELECT statement must also be in the same order.
2. UNION syntax:

|  |
| --- |
| SELECT column\_name(s) FROM table1  UNION  SELECT column\_name(s) FROM table2; |

1. UNION ALL syntax:  
   The UNION operator selects only distinct values by default. To allow duplicate values, use UNION ALL:

|  |
| --- |
| SELECT column\_name(s) FROM table1  UNION ALL  SELECT column\_name(s) FROM table2; |

1. The column names in the result-set are usually equal to the column names in the first SELECT statement.

**SQL UNION Example:**

1. The following SQL statement returns the cities (only distinct values) from both the "Customers" and the "Suppliers" table:
2. Example:

|  |
| --- |
| SELECT City FROM Customers  UNION  SELECT City FROM Suppliers  ORDER BY City; |

|  |
| --- |
| **City** |
| Aachen |
| Albuquerque |
| Anchorage |
| Ann Arbor |
| Annecy |
| Århus |
| Barcelona |
| Barquisimeto |

**SQL UNION ALL Example:**

1. The following SQL statement returns the cities (duplicate values also) from both the "Customers" and the "Suppliers" table:
2. Example:

|  |
| --- |
| SELECT City FROM Customers  UNION ALL  SELECT City FROM Suppliers  ORDER BY City; |

|  |
| --- |
| **City** |
| Aachen |
| Albuquerque |
| Anchorage |
| Ann Arbor |
| Annecy |
| Århus |
| Barcelona |
| Barquisimeto |

**SQL UNION With WHERE:**

1. The following SQL statement returns the German cities (only distinct values) from both the "Customers" and the "Suppliers" table:
2. Example:

|  |
| --- |
| SELECT City, Country FROM Customers  WHERE Country='Germany'  UNION  SELECT City, Country FROM Suppliers  WHERE Country='Germany'  ORDER BY City; |

|  |  |
| --- | --- |
| **City** | **Country** |
| Aachen | Germany |
| Berlin | Germany |
| Brandenburg | Germany |
| Cunewalde | Germany |
| Cuxhaven | Germany |
| Frankfurt | Germany |
| Frankfurt a.M. | Germany |
| Köln | Germany |

**SQL UNION ALL With WHERE:**

1. The following SQL statement returns the German cities (duplicate values also) from both the "Customers" and the "Suppliers" table:
2. Example:

|  |
| --- |
| SELECT City, Country FROM Customers  WHERE Country='Germany'  UNION ALL  SELECT City, Country FROM Suppliers  WHERE Country='Germany'  ORDER BY City; |

|  |  |
| --- | --- |
| **City** | **Country** |
| Aachen | Germany |
| Berlin | Germany |
| Berlin | Germany |
| Brandenburg | Germany |
| Cunewalde | Germany |
| Cuxhaven | Germany |
| Frankfurt | Germany |
| Frankfurt a.M. | Germany |

**Another UNION Example:**

1. The following SQL statement lists all customers and suppliers:
2. Example:

|  |
| --- |
| SELECT 'Customer' AS Type, ContactName, City, Country  FROM Customers  UNION  SELECT 'Supplier', ContactName, City, Country  FROM Suppliers; |

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **ContactName** | **City** | **Country** |
| Customer | Alejandra Camino | Madrid | Spain |
| Customer | Alexander Feuer | Leipzig | Germany |
| Customer | Ana Trujillo | México D.F. | Mexico |
| Customer | Anabela Domingues | São Paulo | Brazil |
| Customer | André Fonseca | Campinas | Brazil |
| Customer | Ann Devon | London | UK |
| Customer | Annette Roulet | Toulouse | France |
| Customer | Antonio Moreno | México D.F. | Mexico |

Notice the "AS Type" above - it is an alias. SQL Aliases are used to give a table or a column a temporary name. An alias only exists for the duration of the query. So, here we have created a temporary column named "Type", that list whether the contact person is a "Customer" or a "Supplier".