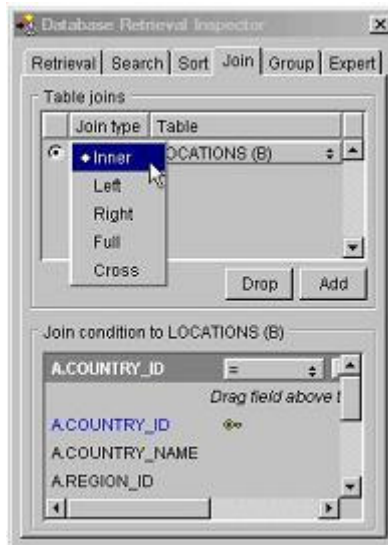


## Joins - how they work

A join is an operation that defines a relationship between two or more tables to determine which records are retrieved or acted upon. What that means is that a join is conditional — similar to a WHERE clause or criteria expression — in that the join specifies which records (rows) are selected in both tables. The type of join significantly impacts which records are retrieved or acted upon.

Scribe's Database Retrieval component supports a number of different types of join operation. The graphical interface provided by Scribe in the Join tab of Database Retrieval Inspector offers the following types of join:



The best way to explain the working of Joins is by example, based on two tables: Customers and Orders.

Customers table

CustomerID	CustomerName
1	George
2	William
3	Vivian
4	Steve

Orders table

OrderID	CustomerID	Amount
12432	1	100.00
65091	1	760.00
70329	2	250.00
15609	3	842.00
89437	5	79.00

### Inner join

Returns all rows from the left table (Customers) that can be joined with rows in the right table (Orders). Rows in Customers that do not have counterparts in Orders are not included in the result.

Example:

```
select CustomerName, OrderID, Amount from Customers join Orders on
Customers.CustomerID = Orders.CustomerID
```

<i>CustomerName</i>	<i>OrderID</i>	<i>Amount</i>
George	12432	100.00
George	65091	760.00
William	70329	250.00
Vivian	15609	842.00

### Left Outer join

Returns all rows from the left table (Customers), joined with rows in the right table (Orders). Rows in Customers that do not have counterparts in Orders are included in the result with NULLs for the fields in Orders table.

Example:

```
select CustomerName, OrderID, Amount from Customers left join Orders on
Customers.CustomerID = Orders.CustomerID
```

<i>CustomerName</i>	<i>OrderID</i>	<i>Amount</i>
George	12432	100.00
George	65091	760.00
William	70329	250.00
Vivian	15609	842.00
Steve	NULL	NULL

### Right Outer join

Returns all rows from the right table (Orders), joined with rows in the left table (Customers). Rows in B that do not have counterparts in Customers are included in the result with NULLs for the fields in Customers table.

Example:

```
select CustomerName, OrderID, Amount from Customers right join Orders on
Customers.CustomerID = Orders.CustomerID
```

<i>CustomerName</i>	<i>OrderID</i>	<i>Amount</i>
George	12432	100.00
George	65091	760.00
William	70329	250.00
Vivian	15609	842.00
NULL	89437	79.00

### Full Outer join

A combination of the left and right outer joins. Any row in one table that does not have a matching key in the other table is joined with a row of NULLs.

Example:

```
select CustomerName, OrderID, Amount from Customers full join Orders on
Customers.CustomerID = Orders.CustomerID
```

<i>CustomerName</i>	<i>OrderID</i>	<i>Amount</i>
George	12432	100.00
George	65091	760.00
William	70329	250.00
Vivian	15609	842.00
Steve	NULL	NULL
NULL	89437	79.00

### Cross join

The Cross join returns what's known as a Cartesian product. This means that the join combines every row from the left table with every row in the right table. The total number of the returned rows is a product of the numbers of rows in two tables.

Example:

```
select CustomerName, OrderID, Amount from Customers cross join Orders
```

<i>CustomerName</i>	<i>OrderID</i>	<i>Amount</i>
George	12432	100.00
George	65091	760.00
George	70329	250.00
George	15609	842.00
George	89437	79.00
William	12432	100.00
William	65091	760.00
William	70329	250.00
William	15609	842.00
William	89437	79.00
Vivian	12432	100.00
Vivian	65091	760.00
Vivian	70329	250.00
Vivian	15609	842.00
Vivian	89437	79.00
Steve	12432	100.00
Steve	65091	760.00
Steve	70329	250.00
Steve	15609	842.00
Steve	89437	79.00