



COMBINACIÓN DE TABLAS

COMBINACIÓN DE TABLAS

¿Qué se entiende por combinación de tablas?

Una combinación es una operación que permite consultar **dos o más tablas** para producir un conjunto de resultados que incorpore filas y columnas de cada una de las tablas en cuestión.

Las tablas se combinan en función de las columnas que son comunes a ambas tablas.

COMBINACIÓN DE TABLAS

El objetivo de la combinación de tablas es proporcionar al usuario datos que le permitan un fácil entendimiento de la información que requiere.

Esta información, por el uso del modelo entidad – relación, se puede encontrar fragmentada en muchas tablas, y al combinarlas, se puede presentar al usuario la información pertinente de una forma más entendible.

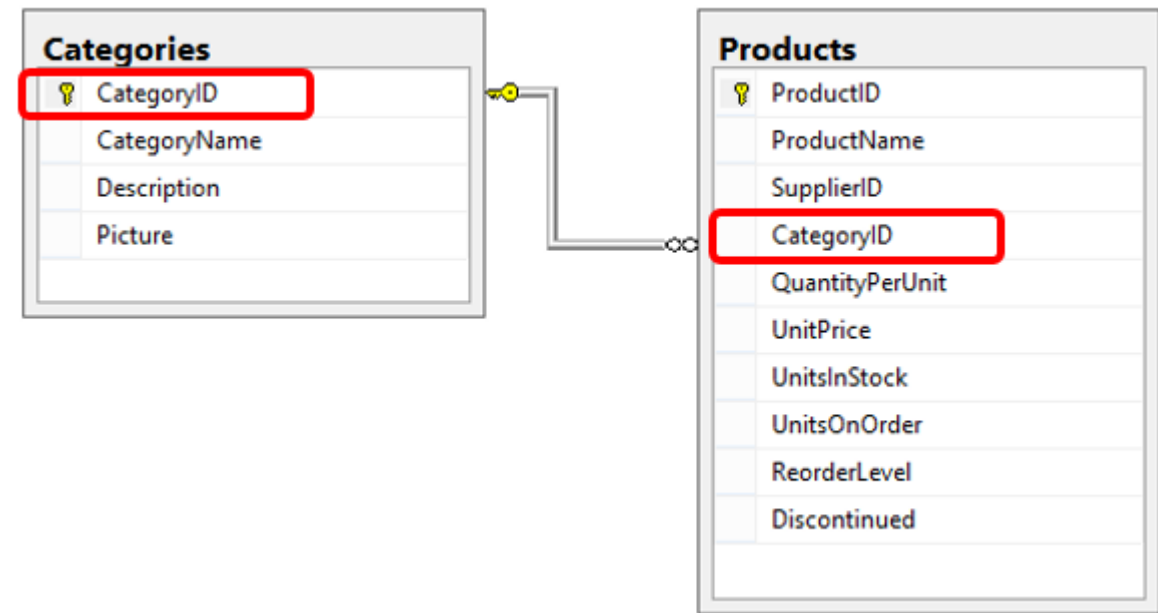
Esta operación es conocida también **como unión o vinculación de tablas.**

COMBINACIÓN DE TABLAS

La combinación de campos de tablas distintas sólo es posible cuando se han definido campos relacionados entre tablas.

Esto es si existe un campo clave primaria en una tabla que aparece como clave foránea en la otra tabla.

La sentencia **JOIN** en SQL permite combinar registros de dos o más tablas en una base de datos relacional.



SENTENCIAS JOIN

En el Lenguaje de Consultas Estructurado (SQL) hay tres tipos de JOIN: **interno, externo y cruzado.**

Combinación interna **INNER JOIN**

Cruzada **CROSS JOIN**

Combinación externa **OUTER JOIN**

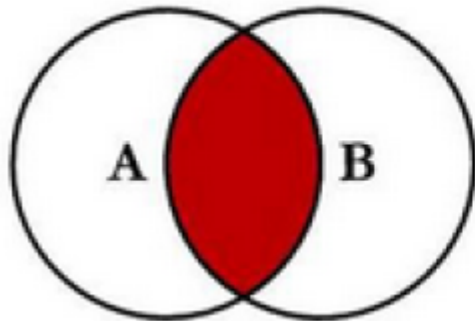
- LEFT OUTER JOIN o LEFT JOIN
- RIGHT OUTER JOIN o RIGHT JOIN
- FULL OUTER JOIN o FULL JOIN

La palabra **OUTER** es opcional y no añade ninguna función

SQL JOINS

INNER JOIN

Se utiliza para mostrar los datos coincidentes entre las tablas de donde se quiere mostrar la información:



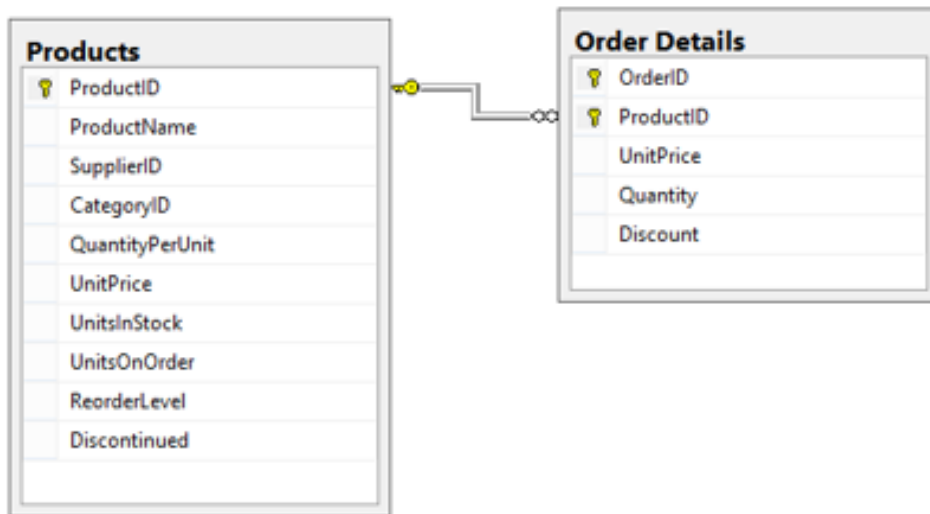
```
SELECT <lista_campos>  
FROM <TablaA A>  
INNER JOIN <TablaB B>  
ON A.Key=B.Key
```

Nota: **ON** se utiliza para colocar los nombres de los campos con los cuales se ha realizado la relación entre las tablas.

SQL JOINS

Ejemplo

Se desea conocer todos los productos que se encuentran en una orden



Para obtener los registros coincidentes en ambas tablas habría que realizar la siguiente consulta:

```
SELECT OrderID, P.ProductID, ProductName
FROM Products P
INNER JOIN [Order Details] OD
ON P.ProductID=OD.ProductID
```

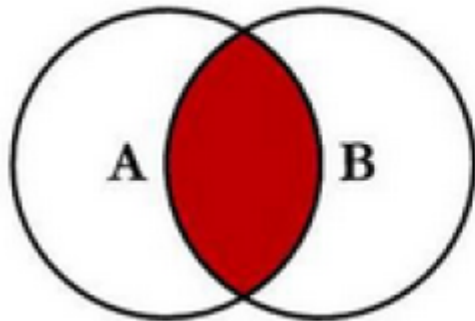
Resultado de la consulta:

	OrderID	ProductID	ProductName
1	10285	1	Chai
2	10294	1	Chai
3	10317	1	Chai
4	10348	1	Chai
5	10354	1	Chai
6	10370	1	Chai
7	10406	1	Chai
8	10413	1	Chai
9	10477	1	Chai
10	10522	1	Chai

SQL JOINS

INNER JOIN

Se utiliza para mostrar los datos coincidentes entre las tablas de donde se quiere mostrar la información:



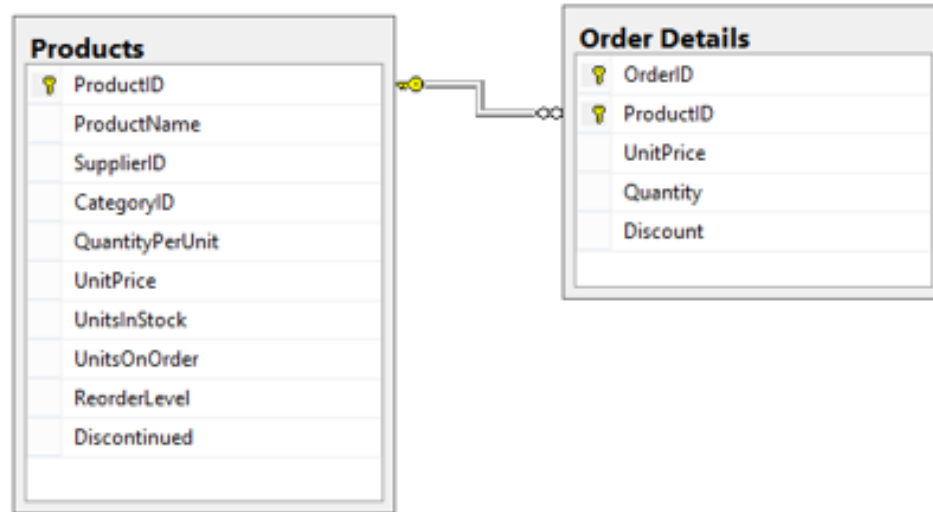
```
SELECT <lista_campos>  
FROM <TablaA A>  
INNER JOIN <TablaB B>  
ON A.Key=B.Key
```

Nota: **ON** se utiliza para colocar los nombres de los campos con los cuales se ha realizado la relación entre las tablas.

SQL JOINS

Ejemplo 1

Se desea conocer todos los productos que se encuentran en una orden



SQL JOINS

Para obtener los registros coincidentes en ambas tablas habría que realizar la siguiente consulta:

```
SELECT OrderID, P.ProductID, ProductName
FROM Products P
INNER JOIN [Order Details] OD
ON P.ProductID=OD.ProductID
```

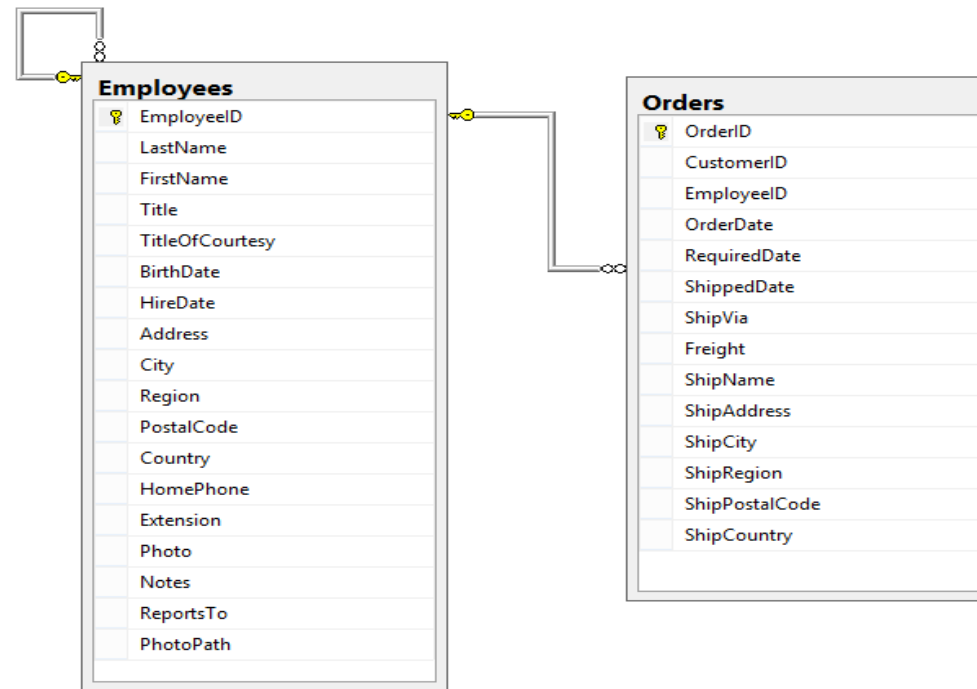
Resultado de la consulta:

	OrderID	ProductID	ProductName
1	10285	1	Chai
2	10294	1	Chai
3	10317	1	Chai
4	10348	1	Chai
5	10354	1	Chai
6	10370	1	Chai
7	10406	1	Chai
8	10413	1	Chai
9	10477	1	Chai
10	10522	1	Chai

SQL JOINS

Ejemplo 2

Se desea conocer los empleados que han atendido una orden y en qué fecha lo hicieron, los registros se deben ordenar por el campo EmployeeID



SQL JOINS

La consulta SQL es:

```
SELECT LastName, Employees.EmployeeID, OrderDate FROM Orders  
INNER JOIN Employees  
ON Orders.EmployeeID=Employees.EmployeeID  
ORDER BY Employees.EmployeeID
```

Resultado de la consulta:

	OrderID	ProductID	ProductName
1	10285	1	Chai
2	10294	1	Chai
3	10317	1	Chai
4	10348	1	Chai
5	10354	1	Chai
6	10370	1	Chai
7	10406	1	Chai
8	10413	1	Chai
9	10477	1	Chai
10	10522	1	Chai

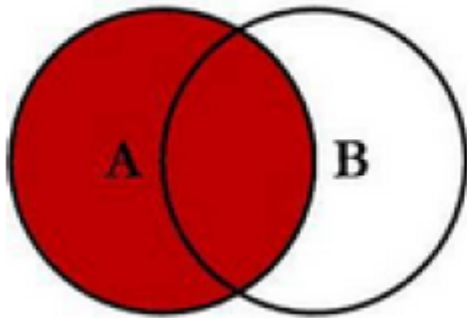
Nota: al campo EmployeeID se le coloca el nombre de la tabla de donde queremos sacar los resultados, ya que el nombre de este campo aparece tanto en la tabla Orders y Employees, y si no se utiliza así da error de nombre ambiguo.

Aquí se muestran las primeras 12 filas de 830 filas en total

SQL JOINS

LEFT JOIN

Muestra los registros de la tabla izquierda más los registros coincidentes con la tabla derecha

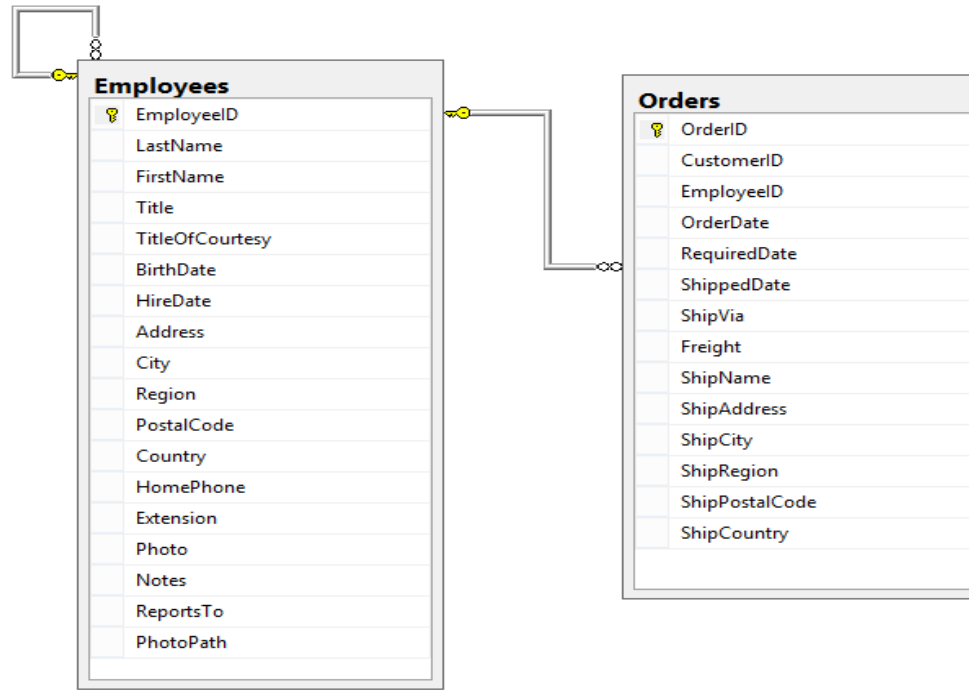


```
SELECT <lista_campos>  
FROM <TablaA A>  
LEFT JOIN <TablaB B>  
ON A.Key=B.Key
```

SQL JOINS

Ejemplo

Se desea conocer que empleados han atendido un pedido independientemente si este lo ha realizado o no



SQL JOINS

La consulta SQL es:

```
SELECT OrderID, E.EmployeeID, Lastname  
FROM Employees E  
LEFT JOIN Orders O  
ON E.EmployeeID=O.EmployeeID
```

Resultado de la consulta:

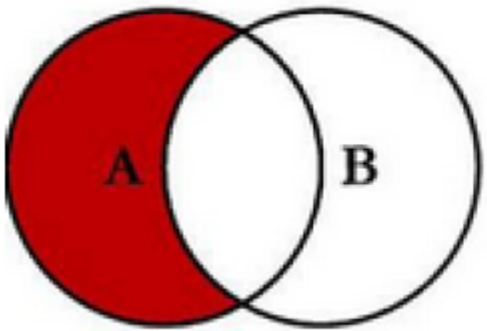
	OrderID	EmployeeID	Lastname
821	10944	6	Suyama
822	10956	6	Suyama
823	10959	6	Suyama
824	10965	6	Suyama
825	10973	6	Suyama
826	10999	6	Suyama
827	11019	6	Suyama
828	11025	6	Suyama
829	11031	6	Suyama
830	11045	6	Suyama
831	NULL	10	Urrutia

En el último registro se observa que en el campo OrderID tiene un valor NULL, lo cual indica que el empleado Urrutia no ha atendido ningún pedido

SQL JOINS

LEFT JOIN (IS NULL)

Muestra los registros de la tabla izquierda menos los registros coincidentes con la tabla derecha

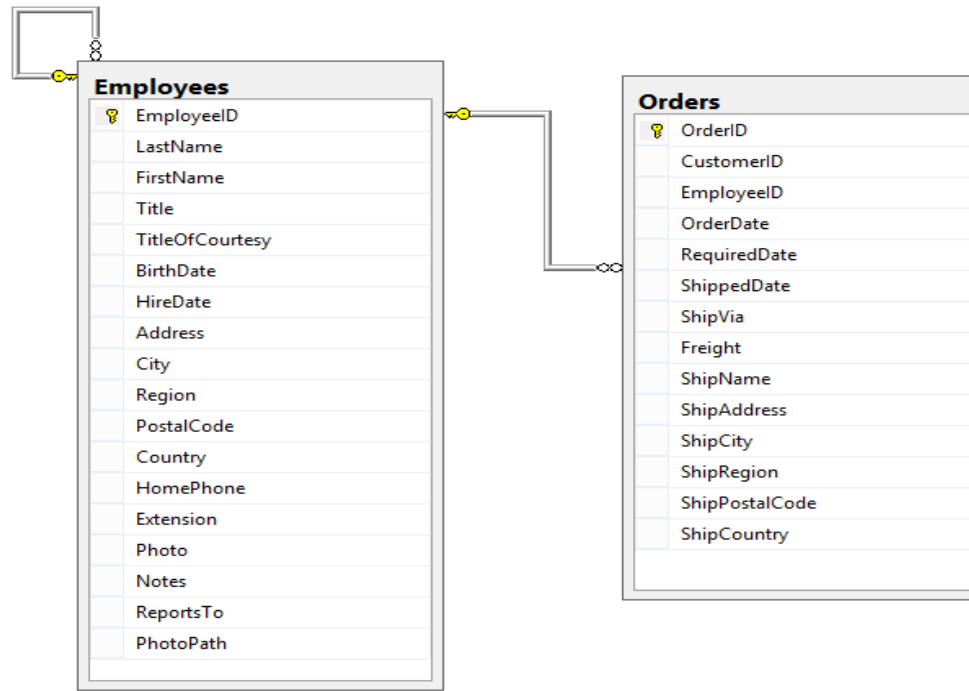


```
SELECT <lista_campos>  
FROM <TablaA A>  
LEFT JOIN <TablaB B>  
ON A.Key=B.Key  
WHERE B.Key IS NULL
```


SQL JOINS

Ejemplo

Se desea conocer los empleados que no han atendido ningún pedido



SQL JOINS

La consulta SQL es:

```
SELECT OrderID, E.EmployeeID, Lastname  
FROM Employees E  
LEFT JOIN Orders O  
ON E.EmployeeID=O.EmployeeID  
WHERE O.EmployeeID IS NULL
```

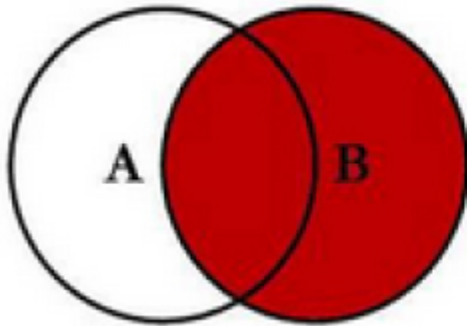
Resultado de la consulta (un solo registro):

	OrderID	EmployeeID	Lastname
1	NULL	10	Umutia

SQL JOINS

RIGHT JOIN

Muestra los registros de la tabla derecha más los registros coincidentes con la tabla izquierda

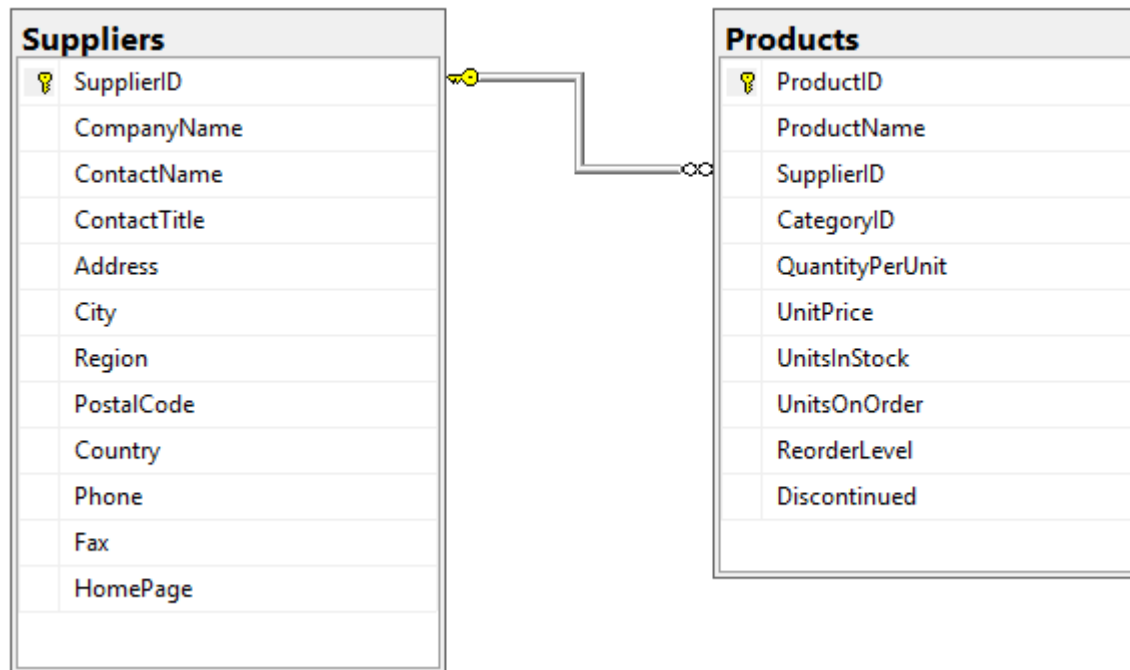


```
SELECT <lista_campos>  
FROM <TablaA A>  
RIGHT JOIN <TablaB B>  
ON A.Key=B.Key
```

SQL JOINS

Ejemplo

Mostrar que productos ofrece cada proveedor independientemente si este lo hace o no



SQL JOINS

La consulta SQL es:

```
SELECT ProductName, CompanyName, ContactName
FROM Products P
RIGHT JOIN Suppliers S
ON P.SupplierID=S.SupplierID
```

Resultado de la consulta:

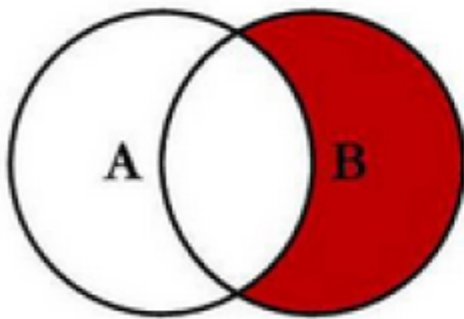
	ProductName	CompanyName	ContactName
74	Raclette Courdavault	Gai pâturage	Eliane Noz
75	Camembert Pierrot	Gai pâturage	Eliane Noz
76	Sirop d'érable	Forêts d'érables	Chantal Goulet
77	Tarte au sucre	Forêts d'érables	Chantal Goulet
78	NULL	Coca Cola	Iñaky Perez

En el último registro se verifica que el proveedor Coca Cola no ha ofrecido ningún producto

SQL JOINS

RIGHT JOIN (IS NULL)

Muestra los registros de la tabla derecha menos los registros coincidentes con la tabla izquierda

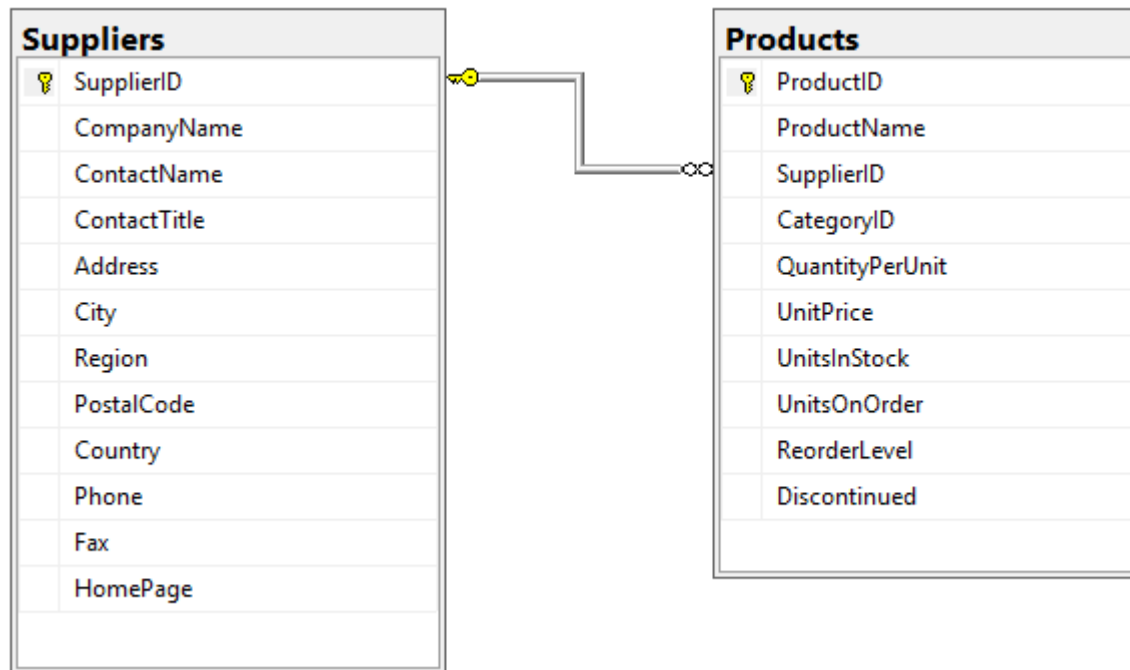


```
SELECT <lista_campos>  
FROM <TablaA A>  
RIGHT JOIN <TablaB B>  
ON A.Key=B.Key  
WHERE A.Key IS NULL
```

SQL JOINS

Ejemplo

Mostrar que proveedor no ha ofrecido productos



SQL JOINS

La consulta SQL es:

```
SELECT ProductName, CompanyName, ContactName  
FROM Products P  
RIGHT JOIN Suppliers S  
ON P.SupplierID=S.SupplierID  
WHERE P.SupplierID IS NULL
```

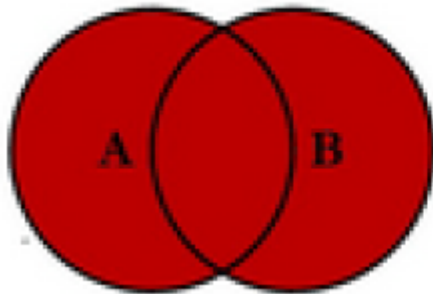
Resultado de la consulta (un solo registro):

	ProductName	CompanyName	ContactName
1	NULL	Coca Cola	Iñaky Perez

SQL JOINS

FULL JOIN

Muestra los registros de la tabla izquierda y la tabla derecha más los registros coincidentes entre ambas

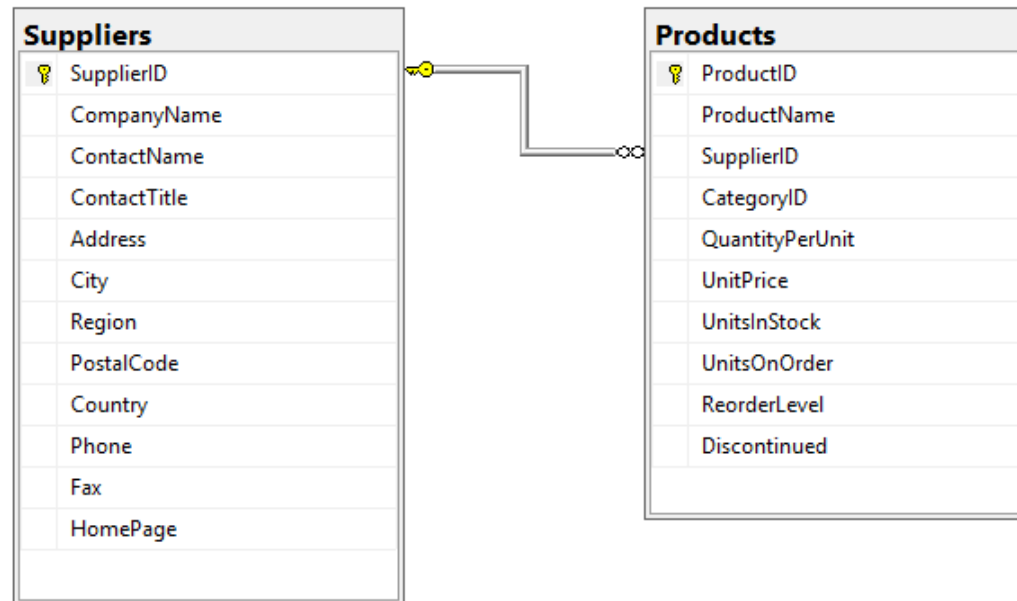


```
SELECT <lista_campos>  
FROM <TablaA A>  
FULL JOIN <TablaB B>  
ON A.Key=B.Key
```

SQL JOINS

Ejemplo

Mostrar los productos que tengan o no asignado un proveedor y los proveedores independientemente si estos han ofrecido o no un producto



SQL JOINS

La consulta SQL es:

```
SELECT ProductName, CompanyName, ContactName
FROM Products P
FULL JOIN Suppliers S
ON P.SupplierID=S.SupplierID
```

Resultado de la consulta:

	ProductName	CompanyName	ContactName
1	Producto X	NULL	NULL
2	Chai	Exotic Liquids	Charlotte Cooper
3	Chang	Exotic Liquids	Charlotte Cooper
4	Aniseed Svrup	Exotic Liquids	Charlotte Cooper

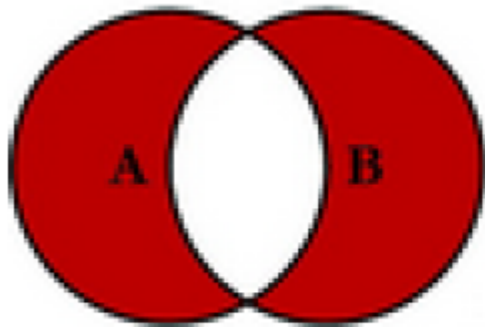
	ProductName	CompanyName	ContactName
75	Raclette Courdavault	Gai pâturage	Eliane Noz
76	Camembert Pierrot	Gai pâturage	Eliane Noz
77	Sirop d'érable	Forêts d'érables	Chantal Goulet
78	Tarte au sucre	Forêts d'érables	Chantal Goulet
79	NULL	Coca Cola	Iñaky Perez

Observar que el primer registro indica que el Producto X no tiene proveedor asignado y en el último registro el Proveedor Coca Cola no ha ofrecido ningún producto.

SQL JOINS

FULL JOIN (IS NULL)

Muestra los registros de la tabla izquierda y la tabla derecha menos los registros coincidentes entre ambas

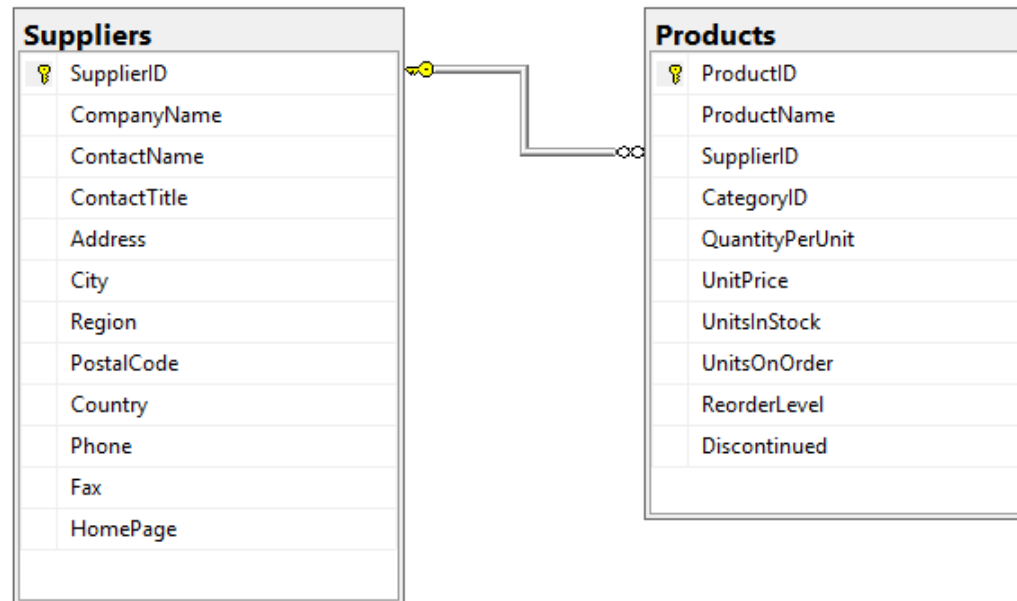


```
SELECT <lista_campos>
FROM <TablaA A>
FULL JOIN <TablaB B>
ON A.Key=B.Key
WHERE A.Key IS NULL OR B.Key IS
NULL
```

SQL JOINS

Ejemplo

Mostrar los productos que no tienen asignado un proveedor y los proveedores que no han ofrecido un producto



SQL JOINS

La consulta SQL es:

```
SELECT ProductName, CompanyName, ContactName
FROM Products P
FULL JOIN Suppliers S
ON P.SupplierID=S.SupplierID
WHERE P.SupplierID IS NULL OR S.SupplierID IS NULL
```

Resultado de la consulta:

	ProductName	CompanyName	ContactName
1	Producto X	NULL	NULL
2	NULL	Coca Cola	Iñaky Perez

SQL JOINS

CROSS JOIN

Una combinación cruzada que no tenga una cláusula WHERE genera el producto cartesiano de las tablas involucradas en la combinación.

El tamaño del conjunto de resultados de un producto cartesiano es igual al número de filas de la primera tabla multiplicado por el número de filas de la segunda tabla.

SQL JOINS

Ejemplo

Ejecutamos las siguientes consultas para conocer la cantidad de filas o registros tienen las siguientes tablas:

```
SELECT * FROM Products
```

Northwind	00:00:00	78 rows
-----------	----------	---------

```
SELECT * FROM Suppliers
```

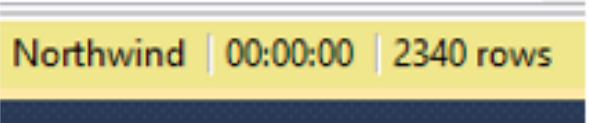
Northwind	00:00:00	30 rows
-----------	----------	---------

SQL JOINS

Ejemplo

Ahora ejecutamos la siguiente consulta:

```
SELECT ProductName, CompanyName, ContactName  
FROM Products P  
CROSS JOIN Suppliers S
```



Northwind | 00:00:00 | 2340 rows

Como resultado tenemos 2340 filas o registros, ya que si multiplicamos las 78 filas de la primera tabla por las 30 filas de la segunda obtenemos ese resultado

SQL JOINS

Sin embargo, si se agrega una cláusula WHERE, la combinación cruzada se comporta como una combinación interna (INNER JOIN)

```
SELECT ProductName, CompanyName, ContactName
FROM Products P
CROSS JOIN Suppliers S
WHERE P.SupplierID=S.SupplierID
```

Obtenemos el mismo resultado al ejecutar la siguiente consulta:

```
SELECT ProductName, CompanyName, ContactName
FROM Products P
INNER JOIN Suppliers S
ON P.SupplierID=S.SupplierID
```