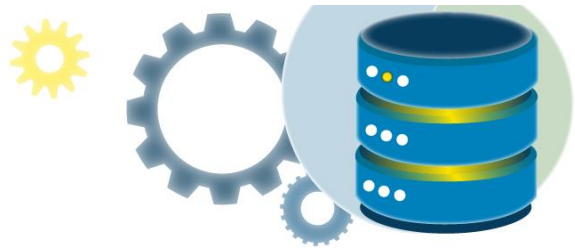


Consultas SQL con condicionales

En este tutorial usaremos la conocida base de datos de muestra Northwind.

En este documento usted encontrará:

UNIONES SQL (JOINS)

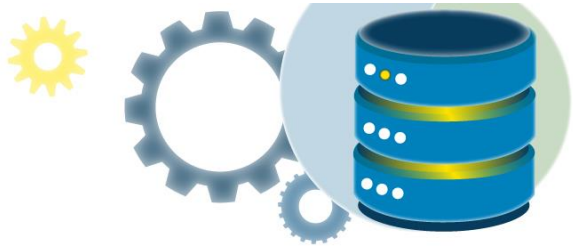
- Unión INNER JOIN
- Unión LEFT JOIN
- Unión RIGHT OUTER JOIN
- Unión FULL OUTER JOIN
- Unión Self Join
- El operador SQL UNION

La instrucción GROUP BY de SQL

La cláusula HAVING de SQL

UNIONES SQL (JOINS)

Una cláusula JOIN se utiliza para combinar filas de dos o más tablas, según una columna relacionada entre ellas.



Veamos una selección de la tabla "Pedidos":

OrderID	CustomerID	OrderDate
10308	2	1996-09-18
10309	37	1996-09-19
10310	77	1996-09-20

Luego, observe una selección de la tabla "Clientes":

CustomerID	CustomerName	ContactName	Country
1	Alfreds Futterkiste	Maria Anders	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mexico

Observe que la columna "CustomerID" en la tabla "Pedidos" se refiere al "CustomerID" en la tabla "Clientes". La relación entre las dos tablas anteriores es la columna "CustomerID".

Luego, podemos crear la siguiente declaración SQL (que contiene un `INNER JOIN`), que selecciona registros que tienen valores coincidentes en ambas tablas:

Ejemplo

```
SELECT Orders.OrderID, Customers.CustomerName, Orders.OrderDate
FROM Orders
INNER JOIN Customers ON Orders.CustomerID=Customers.CustomerID;
```

y producirá, algo como esto:

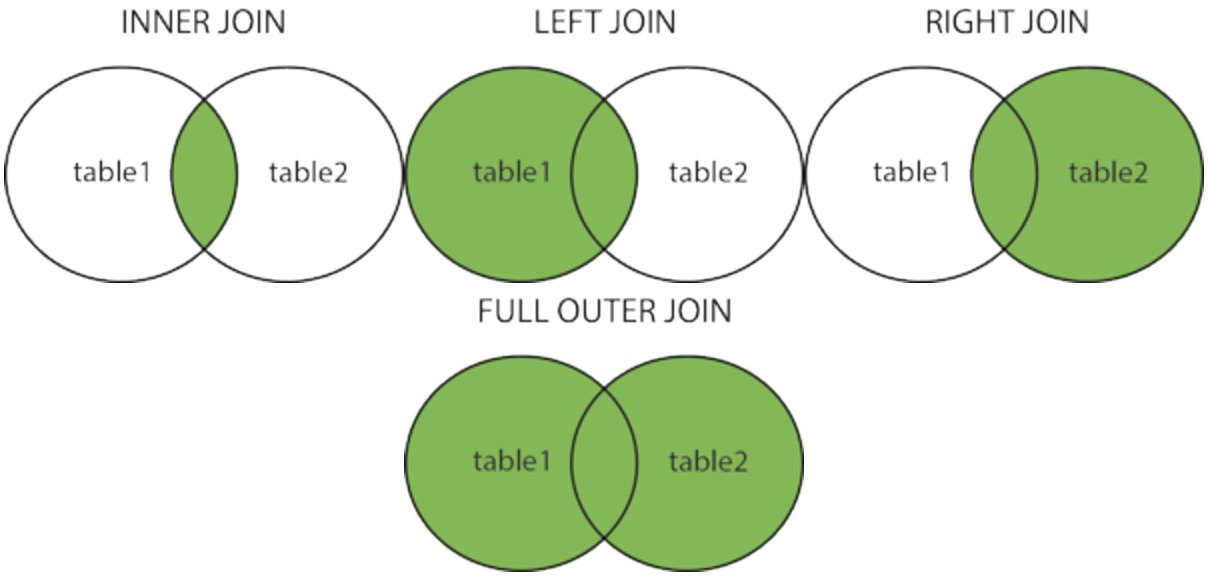
OrderID	CustomerName	OrderDate
10308	Ana Trujillo Emparedados y helados	9/18/1996
10365	Antonio Moreno Taquería	11/27/1996
10383	Around the Horn	12/16/1996

10355	Around the Horn	11/15/1996
10278	Berglunds snabbköp	8/12/1996

Diferentes tipos de JOIN SQL

Estos son los diferentes tipos de JOIN en SQL:

- `(INNER) JOIN`: Devuelve registros que tienen valores coincidentes en ambas tablas.
- `LEFT (OUTER) JOIN`: Retorna todos los registros de la tabla de la izquierda y los registros coincidentes de la tabla de la derecha.
- `RIGHT (OUTER) JOIN`: Devuelve todos los registros de la tabla de la derecha y los registros coincidentes de la tabla de la izquierda.
- `FULL (OUTER) JOIN`: Selecciona todos los registros cuando hay una coincidencia en la tabla izquierda o derecha.

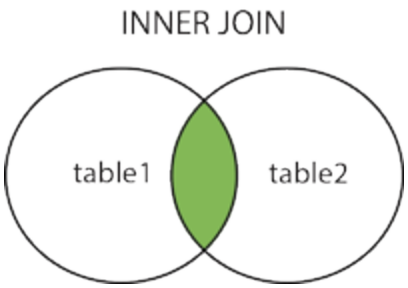


Unión INNER JOIN

La unión INNER JOIN selecciona, registros que tienen valores coincidentes en ambas tablas.

Sintaxis de INNER JOIN

```
SELECT column_name(s)
FROM table1
INNER JOIN table2
ON table1.column_name = table2.column_name;
```



Base de datos de demostración

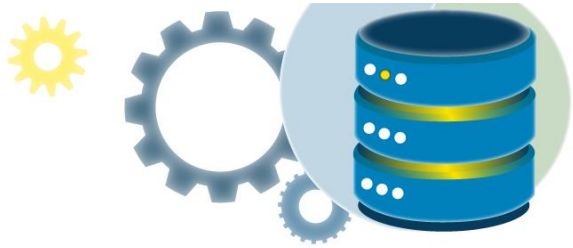
A continuación, se muestra una selección de la tabla "Pedidos":

OrderID	CustomerID	EmployeeID	OrderDate	ShipperID
10308	2	7	1996-09-18	3
10309	37	3	1996-09-19	1
10310	77	8	1996-09-20	2

Y una elección de la tabla "Clientes":

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución 2222	México D.F.	05021	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	05023	Mexico

La siguiente declaración SQL selecciona todos los pedidos con información del cliente:



Ejemplo

```
SELECT Orders.OrderID, Customers.CustomerName
FROM Orders
INNER JOIN Customers ON Orders.CustomerID =
Customers.CustomerID;
```

Nota: La unión INNER JOIN selecciona todas las filas de ambas tablas siempre que haya una coincidencia entre las columnas. Si hay registros en la tabla "Pedidos" que no tienen coincidencias en "Clientes", ¡estos pedidos no se mostrarán!

Unión de tres tablas

La siguiente declaración SQL selecciona todos los pedidos con información del cliente y del remitente:

```
SELECT Orders.OrderID, Customers.CustomerName,
Shippers.ShipperName
FROM ((Orders
INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID)
INNER JOIN Shippers ON Orders.ShipperID = Shippers.ShipperID);
```

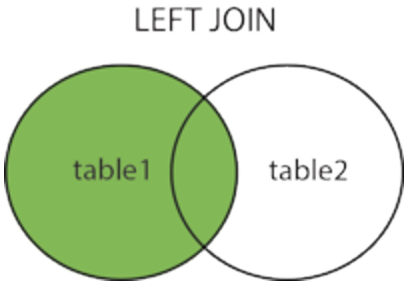
Unión LEFT JOIN

La unión LEFT JOIN clave devuelve todos los registros de la tabla izquierda (tabla1) y los registros coincidentes de la tabla derecha (tabla2). El resultado es 0 registros del lado derecho, si no hay coincidencia.

Sintaxis de LEFT JOIN

```
SELECT column_name(s)
FROM table1
LEFT JOIN table2
ON table1.column_name = table2.column_name;
```

Nota: En algunas bases de datos, LEFT JOIN se llama LEFT OUTER JOIN, como en SQL Server.



Base de datos de demostración

En este tutorial usaremos la conocida base de datos de muestra Northwind.

A continuación, se muestra una selección de la tabla "Clientes":

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución 2222	México D.F.	05021	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	05023	Mexico

Y una selección de la tabla "Pedidos":

OrderID	CustomerID	EmployeeID	OrderDate	ShipperID
10308	2	7	1996-09-18	3
10309	37	3	1996-09-19	1
10310	77	8	1996-09-20	2

Ejemplo

La siguiente declaración SQL seleccionará a todos los clientes y cualquier pedido que puedan tener:

```
SELECT Customers.CustomerName, Orders.OrderID
FROM Customers
LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID
ORDER BY Customers.CustomerName;
```

Nota: La unión LEFT JOIN clave devuelve todos los registros de la tabla de la izquierda (Clientes), incluso si no hay coincidencias en la tabla de la derecha (Pedidos).

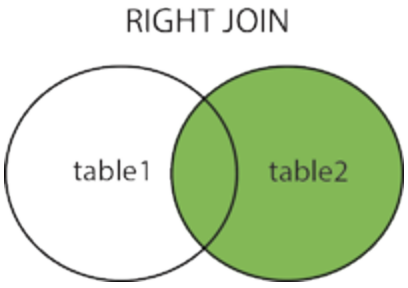
Unión RIGHT OUTER JOIN

La unión RIGHT OUTER JOIN devuelve todos los registros de la tabla derecha (tabla2) y los registros coincidentes de la tabla izquierda (tabla1). El resultado es 0 registros del lado izquierdo, si no hay coincidencia.

Sintaxis de RIGHT OUTER JOIN

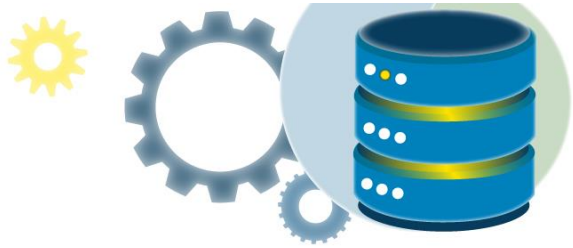
```
SELECT column_name(s)
FROM table1
RIGHT OUTER JOIN table2
ON table1.column_name = table2.column_name;
```

Nota: En algunas bases de datos RIGHT OUTER JOIN se llama RIGHT JOIN.



Base de datos de demostración

En este tutorial usaremos la conocida base de datos de muestra Northwind.



A continuación, se muestra una selección de la tabla "Pedidos":

OrderID	CustomerID	EmployeeID	OrderDate	ShipperID
10308	2	7	1996-09-18	3
10309	37	3	1996-09-19	1
10310	77	8	1996-09-20	2

Y una selección de la tabla "Empleados":

EmployeeID	LastName	FirstName	BirthDate	Photo
1	Davolio	Nancy	12/8/1968	EmpID1.pic
2	Fuller	Andrew	2/19/1952	EmpID2.pic
3	Leverling	Janet	8/30/1963	EmpID3.pic

Ejemplo de SQL RIGHT OUTER JOIN

La siguiente instrucción SQL devolverá a todos los empleados y cualquier pedido que hayan realizado:

```
SELECT Orders.OrderID, Employees.LastName, Employees.FirstName
FROM Orders
RIGHT OUTER JOIN Employees ON Orders.EmployeeID =
Employees.EmployeeID
ORDER BY Orders.OrderID;
```

Nota: La unión RIGHT OUTER JOIN devuelve todos los registros de la tabla de la derecha (Empleados), incluso si no hay coincidencias en la tabla de la izquierda (Pedidos).

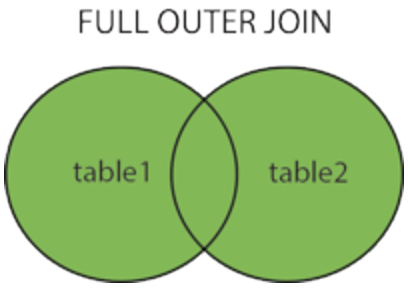
Unión FULL OUTER JOIN

La unión FULL OUTER JOIN retorna todos los registros cuando hay una coincidencia en los registros de la tabla izquierda (tabla1) o derecha (tabla2).

Consejo: FULL OUTER JOIN y FULL JOIN son lo mismo.

Sintaxis de FULL OUTER JOIN

```
SELECT column_name(s)
FROM table1
FULL OUTER JOIN table2
ON table1.column_name = table2.column_name
WHERE condition;
```

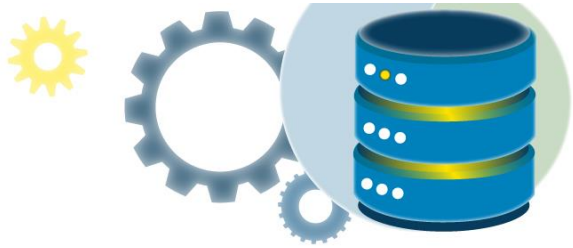


Nota: FULL OUTER JOIN potencialmente puede devolver, conjuntos de resultados muy grandes.

Base de datos de demostración

A continuación, se muestra una selección de la tabla "Clientes":

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución 2222	México D.F.	05021	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	05023	Mexico



Y una selecciónclasificación de la tabla "Pedidos":

OrderID	CustomerID	EmployeeID	OrderDate	ShipperID
10308	2	7	1996-09-18	3
10309	37	3	1996-09-19	1
10310	77	8	1996-09-20	2

Ejemplo

La siguiente declaración SQL selecciona todos los clientes y todos los pedidos:

```
SELECT Customers.CustomerName, Orders.OrderID
FROM Customers
FULL OUTER JOIN Orders ON Customers.CustomerID=Orders.Custom
erID
ORDER BY Customers.CustomerName;
```

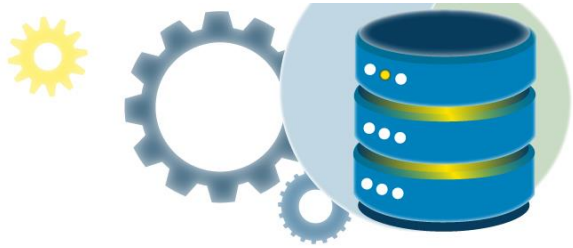
Una selección del conjunto de resultados puede verse así:

CustomerName	OrderID
Alfreds Futterkiste	*Null*
Ana Trujillo Emparedados y helados	10308
Antonio Moreno Taquería	*Null*

Nota: La unión FULL OUTER JOIN devuelve todos los registros coincidentes de ambas tablas, independientemente de que la otra tabla coincida o no. Por lo tanto, si hay filas en "Clientes" que no tienen coincidencias en "Pedidos", o si hay filas en "Pedidos" que no tienen coincidencias en "Clientes", esas filas también se enumerarán.

Unión Self Join

Una autocombinación es una combinación normal, pero la tabla se une a sí misma.



Sintaxis de Self Join

```
SELECT column_name(s)
FROM table1 T1, table1 T2
WHERE condition;
```

T1 y *T2* son alias de tabla diferentes para la misma tabla.

Base de datos de demostración

A continuación, se muestra una selección de la tabla "Clientes":

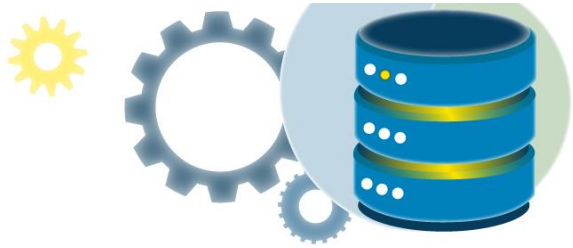
CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución 2222	México D.F.	05021	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	05023	Mexico

Ejemplo

La siguiente declaración SQL coincide con los clientes que son de la misma ciudad:

Ejemplo

```
SELECT A.CustomerName AS CustomerName1,
B.CustomerName AS CustomerName2, A.City
FROM Customers A, Customers B
WHERE A.CustomerID <> B.CustomerID
AND A.City = B.City
ORDER BY A.City;
```



El operador SQL UNION

El operador UNION se utiliza para combinar el conjunto de resultados de dos o más SELECT declaraciones.

- Cada sentencia SELECT dentro de UNION debe tener el mismo número de columnas.
- Las columnas también, deben tener tipos de datos similares.
- Las columnas de cada SELECT también, deben estar en el mismo orden.

Sintaxis UNION

```
SELECT column_name(s) FROM table1
UNION
SELECT column_name(s) FROM table2;
```

Sintaxis de UNION ALL

El operador UNION selecciona solo valores distintos de forma predeterminada. Para permitir valores duplicados, use UNION ALL:

```
SELECT column_name(s) FROM table1
UNION ALL
SELECT column_name(s) FROM table2;
```

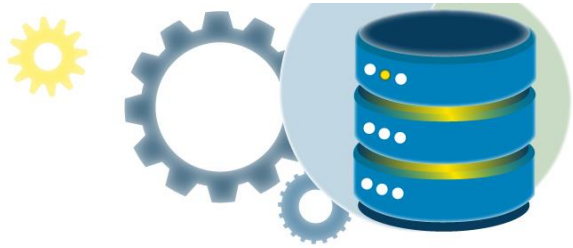
Nota: Los nombres de las columnas en el conjunto de resultados suelen ser iguales, a los nombres de las columnas en la primera sentencia SELECT.

Ejemplo

La siguiente instrucción SQL devuelve las ciudades (solo valores distintos) tanto de la tabla "Clientes" como de la tabla "Proveedores":

```
SELECT City FROM Customers
UNION
SELECT City FROM Suppliers
ORDER BY City;
```

Nota: Si algunos clientes o proveedores tienen la misma ciudad, cada ciudad solo aparecerá una vez, porque UNION selecciona solo valores distintos. Use UNION ALL para seleccionar también, valores duplicados.



Ejemplo de SQL UNION ALL

La siguiente instrucción SQL devuelve las ciudades (también valores duplicados) de las tablas "Clientes" y "Proveedores":

```
SELECT City FROM Customers
UNION ALL
SELECT City FROM Suppliers
ORDER BY City;
```

UNION con WHERE

La siguiente instrucción SQL devuelve las ciudades alemanas (solo valores distintos) de las tablas "Clientes" y "Proveedores":

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

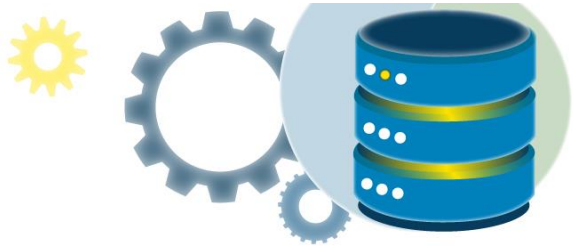
UNION ALL con WHERE

La siguiente instrucción SQL devuelve las ciudades alemanas (también, valores duplicados) de las tablas "Clientes" y "Proveedores":

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

Otro ejemplo de UNION

La siguiente declaración SQL enumera todos los clientes y proveedores:



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

Observe el "Tipo AS" de arriba: es un alias. Los alias SQL se utilizan para dar un nombre temporal a una tabla o columna. Un alias solo existe mientras dura la consulta. Entonces, aquí hemos creado una columna temporal llamada "Tipo", que enumera, si la persona de contacto es un "Cliente" o un "Proveedor".

La instrucción GROUP BY de SQL

La cláusula GROUP BY agrupa las filas que tienen los mismos valores en filas de resumen, como "encontrar el número de clientes en cada país".

La cláusula GROUP BY se utiliza a menudo con funciones de agregado (COUNT(), MAX(), MIN(), SUM(), AVG()) a grupo el conjunto de resultados de una o más columnas.

Sintaxis de Group by

```
SELECT column_name(s)
FROM table_name
WHERE condition
GROUP BY column_name(s)
ORDER BY column_name(s);
```

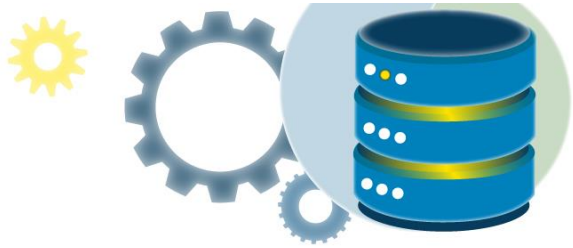
Ejemplo

La siguiente declaración SQL enumera el número de clientes en cada país:

```
SELECT COUNT(CustomerID), Country
FROM Customers
GROUP BY Country;
```

La siguiente declaración SQL enumera el número de clientes en cada país, ordenados de mayor a menor:

```
SELECT COUNT(CustomerID), Country
FROM Customers
GROUP BY Country
ORDER BY COUNT(CustomerID) DESC;
```



GROUP BY con JOIN

La siguiente declaración SQL enumera el número de pedidos enviados por cada remitente:

```
SELECT Shippers.ShipperName, COUNT(Orders.OrderID) AS NumberOfOrd
ers FROM Orders
LEFT JOIN Shippers ON Orders.ShipperID = Shippers.ShipperID
GROUP BY ShipperName;
```

La cláusula HAVING de SQL

La cláusula HAVING se agregó a SQL porque WHERE no se puede usar con funciones agregadas.

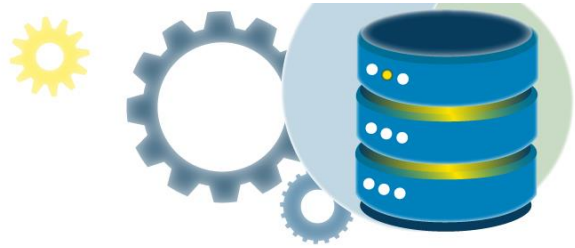
Sintaxis de HAVING

```
SELECT column_name(s)
FROM table_name
WHERE condition
GROUP BY column_name(s)
HAVING condition
ORDER BY column_name(s);
```

Ejemplos de HAVING

La siguiente declaración SQL enumera el número de clientes en cada país. Incluya solo países con más de 5 clientes:

```
SELECT COUNT(CustomerID), Country
FROM Customers
GROUP BY Country
HAVING COUNT(CustomerID) > 5;
```



La siguiente declaración SQL enumera la cantidad de clientes en cada país, ordenados de mayor a menor (solo incluye países con más de 5 clientes):

```
SELECT COUNT(CustomerID), Country
FROM Customers
GROUP BY Country
HAVING COUNT(CustomerID) > 5
ORDER BY COUNT(CustomerID) DESC;
```

La siguiente declaración SQL enumera los empleados que han registrado más de 10 pedidos:

```
SELECT Employees.LastName, COUNT(Orders.OrderID) AS NumberOf
Orders
FROM (Orders
INNER JOIN Employees ON Orders.EmployeeID =
Employees.EmployeeID)
GROUP BY LastName
HAVING COUNT(Orders.OrderID) > 10;
```

La siguiente declaración SQL enumera si los empleados "Davolio" o "Fuller" han registrado más de 25 pedidos:

```
SELECT Employees.LastName, COUNT(Orders.OrderID) AS NumberOf
Orders
FROM Orders
INNER JOIN Employees ON Orders.EmployeeID =
Employees.EmployeeID
WHERE LastName = 'Davolio' OR LastName = 'Fuller'
GROUP BY LastName
HAVING COUNT(Orders.OrderID) > 25;
```