Consultas combinadas. JOINS

Consultas combinadas.

Habitualmente cuando necesitamos recuperar la información de una base de datos nos encontramos con que dicha información se encuentra repartida en varias tablas, referenciadas a través de varios códigos. De este modo si tuvieramos una tabla de ventas con un campo cliente, dicho campo contendría el código del cliente de la tabla de cliente.

Sin embargo está forma de almacenar la información no resulta muy util a la hora de consultar los datos. **SQL** nos proporciona una forma facil de mostrar la información repartida en varias tablas, las **consultas combinadas** o **JOINS.**

Las consultas combinadas pueden ser de tres tipos:

- Combinación interna
- Combinación externa
- Uniones

[arriba]

Combinación interna.

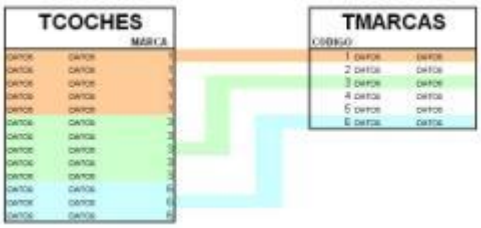
La combinación interna nos permite mostrar los datos de dos o más tablas a través de una condición **WHERE.**

Si recordamos los ejemplos de los capitulos anteriores tenemos una tabla de coches, en la que tenemos referenciada la marca a través del código de marca. Para realizar la consulta combinada entre estas dos tablas debemos escribir una consulta **SELECT** en cuya claúsula **FROM** escribiremos el nombre de las dos tablas, separados por comas, y una condición **WHERE** que obligue a que el código de marca de la tabla de coches sea igual al código de la tabla de marcas.

Lo más sencillo es ver un ejemplo directamente:

```
SELECT tCoches.matricula,
       tMarcas.marca,
       tCoches.modelo,
       tCoches.color,
       tCoches.numero_kilometros,
       tCoches.num_plazas
FROM tCoches, tMarcas
WHERE tCoches.marca = tMarcas.codigo
```

La misma consulta de forma "visual" ...



Demonos cuenta que hemos antepuesto el nombre de cada tabla a el nombre del campo, esto no es obligatorio si los nombres de campos no se repiten en las tablas, pero es acondajable para evitar conflictos de nombres entre campos. Por ejemplo, si para referirnos al campo marca no anteponemos el nombre del campo la base de datos no sabe si queremos el campo marca de la tabla tCoches, que contiene el código de la marca, o el campo marca de la tabla tMarcas, que contiene el nombre de la marca.

Otra opción es utilizar la cláusula **INNER JOIN**. Su sintaxis es identica a la de una consulta **SELECT** habitual, con la particularidad de que én la cláusula **FROM** sólo aparece una tabla o vista, añadiendose el resto de tablas a través de cláusulas **INNER JOIN** .

SELECT [**ALL | DISTINCT**]
<*nombre_campo*> [{,<*nombre_campo*>}]
FROM <*nombre_tabla*>
[{**INNER JOIN** <*nombre_tabla*> **ON** <*condicion_combinacion>}]*
[**WHERE** <*condicion*> [{ **AND**|**OR** <condicion>}]]
[**GROUP BY** <*nombre_campo*> [{,<*nombre_campo* >}]]
[**HAVING** <*condicion*>[{ **AND**|**OR** <condicion>}]]
[**ORDER BY** <*nombre_campo*>|<indice_campo> [**ASC** | **DESC**]
[{,<*nombre_campo*>|<indice_campo> [**ASC** | **DESC**]}]]

El ejemplo anterior escrito utilizando la clausula **INNER JOIN** quedaria de la siguiente manera:

```
SELECT tCoches.matricula,
    tMarcas.marca,
    tCoches.modelo,
    tCoches.color,
    tCoches.numero_kilometros,
    tCoches.num_plazas
FROM tCoches
INNER JOIN tMarcas ON tCoches.marca = tMarcas.codigo
```

La cláusula **INNER JOIN** permite separar completamente las condiciones de combinación con otros criterios, cuando tenemos consultas que combinan nueve o diez tablas esto realmente se agradece. Sin embargo muchos programadores no son amigos de la cláusula **INNER JOIN**, la razón es que uno de los principales gestores de bases de datos, **ORACLE**, no la soportaba. Si nuestro porgrama debia trabajar sobre bases de datos **ORACLE** no podiamos utilizar **INNER JOIN**. **A partir de la version ORACLE 9i oracle soporta la cláusula INNER JOIN**.

[arriba]

Combinación Externa

La combinación interna es excluyente. Esto quiere decir que si un registro no cumple la condición de combinación no se incluye en los resultados. De este modo en el ejemplo anterior si un coche no tiene grabada la marca no se devuelve en mi consulta.

Según la naturaleza de nuestra consulta esto puede ser una ventaja , pero en otros casos significa un serio problema. Para modificar este comportamiento SQL pone a nuestra disposición la combinación externa. La combinación externa no es excluyente.

La sintaxis es muy parecida a la combinación interna,

SELECT [**ALL | DISTINCT**]
<*nombre_campo*> [{,<*nombre_campo*>}]
FROM <*nombre_tabla*>
[{**LEFT|RIGHT OUTER JOIN** <*nombre_tabla*> **ON** <*condicion_combinacion*>}]
[**WHERE** <*condicion*> [{ **AND**|**OR** <condicion>}]]
[**GROUP BY** <*nombre_campo*> [{,<*nombre_campo* >}]]
[**HAVING** <*condicion*>[{ **AND**|**OR** <condicion>}]]
[**ORDER BY** <*nombre_campo*>|<indice_campo> [**ASC** | **DESC**]
[{,<*nombre_campo*>|<indice_campo> [**ASC** | **DESC**]}]]

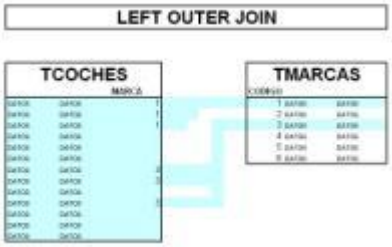
La combinación externa puede ser diestra o siniestra, **LEFT OUTER JOIN** o **RIGHT OUTER JOIN.** Con **LEFT OUTER JOIN** obtenemos todos los registros de en la tabla que situemos a la izquierda de la clausula **JOIN,** mientras que con **RIGHT OUTER JOIN** obtenmos el efecto contrario.

Como mejor se ve la combinación externa es con un ejemplo.

```
SELECT tCoches.matricula,
    tMarcas.marca,
    tCoches.modelo,
    tCoches.color,
    tCoches.numero_kilometros,
    tCoches.num_plazas
FROM tCoches
LEFT OUTER JOIN tMarcas ON tCoches.marca = tMarcas.codigo
```

Esta consulta devolverá todos los registros de la tabla tCoches, independientemente de que tengan marca o no. En el caso de que el coche no tenga marca se devolverá el valor **null** para los campos de la tabla tMarcas.

Visualmente (la consulta devuelve los datos en azul) ...



El mismo ejemplo con RIGHT OUTER JOIN.

```
SELECT tCoches.matricula,
    tMarcas.marca,
    tCoches.modelo,
    tCoches.color,
    tCoches.numero_kilometros,
    tCoches.num_plazas
FROM tCoches
RIGHT OUTER JOIN tMarcas ON tCoches.marca = tMarcas.codigo
```

Esta consulta devolverá los registros de la tabla tCoches que tengan marca relacionada y todos los registros de la tabla tMarcas, tengan algún registro en tCoches o no.

Visualmente (la consulta devuelve los datos en azul) ...

[arriba]

Union

La cláusula **UNION** permite unir dos o más conjuntos de resultados en uno detras del otro como si se tratase de una única tabla. De este modo podemos obtener los registros de mas de una tabla "unidos".

La sintaxis corresponde a la de varias **SELECT** unidas a través de **UNION**, como se muestra a continuación:

```
SELECT [ALL | DISTINCT ]
        <nombre_campo> [{,<nombre_campo>}]
FROM <nombre_tabla>
```

[{**LEFT|RIGHT OUTER JOIN** *<nombre_tabla>* **ON** *<condicion_combinacion>}]*
[**WHERE** <*condicion*> [{ **AND**|**OR** <condicion>}]]
[**GROUP BY** <*nombre_campo*> [{,<*nombre_campo* >}]]
[**HAVING** <*condicion*>[{ **AND**|**OR** <condicion>}]]
{
UNION [ALL | DISTINCT]
SELECT [**ALL | DISTINCT**]
<*nombre_campo*> [{,<*nombre_campo*>}]
FROM <*nombre_tabla*>
[{**LEFT|RIGHT OUTER JOIN** *<nombre_tabla>* **ON** *<condicion_combinacion>}]*
[**WHERE** <*condicion*> [{ **AND**|**OR** <condicion>}]]
[**GROUP BY** <*nombre_campo*> [{,<*nombre_campo* >}]]
[**HAVING** <*condicion*>[{ **AND**|**OR** <condicion>}]]
}
[**ORDER BY** <*nombre_campo*>|<indice_campo> [**ASC** | **DESC**]
[{,<*nombre_campo*>|<indice_campo> [**ASC** | **DESC**]}]]

Para utilizar la clausula **UNION** debemos cumplir una serie de normas.

- Las consultas a unir deben tener el mismo número campos, y además los campos deben ser del mismo tipo.
- Sólo puede haber una única clausula **ORDER BY** al final de la sentencia **SELECT**.

El siguiente ejemplo muestra el uso de **UNION**

```
SELECT tCoches.matricula,
    tMarcas.marca,
    tCoches.modelo,
    tCoches.color,
    tCoches.numero_kilometros,
    tCoches.num_plazas
FROM tCoches
INNER JOIN tMarcas ON tCoches.marca = tMarcas.codigo
UNION
SELECT tMotos.matricula,
    tMarcas.marca,
    tMotos.modelo,
    tMotos.color,
    tMotos.numero_kilometros,
    0
FROM tMotos
INNER JOIN tMarcas ON tMotos.marca = tMarcas.codigo;
```

Puede observarse el uso de la constante cero en la segunda lista de selección para hacer coincidir el número y tipo de
campos que devuelve la consulta UNION.