

Las consultas Multitabla

Introducción

En este tema vamos a estudiar las **consultas multitabla** llamadas así porque están **basadas en más de una tabla**.

El SQL de Microsoft Jet 4.x soporta dos grupos de consultas multitabla:

- la **unión** de tablas
- la **composición** de tablas

La unión de tablas

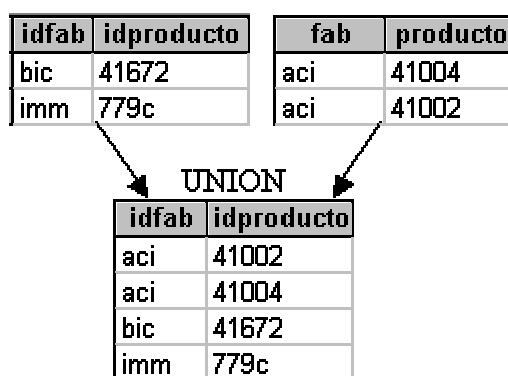
Esta operación se utiliza cuando tenemos **dos tablas** con las **mismas columnas** y queremos obtener una **nueva tabla** con las **filas de la primera y las filas de la segunda**. En este caso la tabla resultante tiene las mismas columnas que la primera tabla (que son las mismas que las de la segunda tabla).

Por ejemplo tenemos una tabla de libros nuevos y una tabla de libros antiguos y queremos una lista con todos los libros que tenemos. En este caso las dos tablas tienen las mismas columnas, lo único que varía son las filas, además queremos obtener una lista de libros (las columnas de una de las tablas) con las filas que están tanto en libros nuevos como las que están en libros antiguos, en este caso utilizaremos este tipo de operación.

Cuando hablamos de tablas pueden ser **tablas reales** almacenadas en la base de datos o **tablas lógicas** (resultados de una consulta), esto nos permite utilizar la operación con más frecuencia ya que pocas veces tenemos en una base de datos tablas idénticas en cuanto a columnas. El **resultado** es siempre una **tabla lógica**.

Por ejemplo queremos en un sólo listado los productos cuyas existencias sean iguales a cero y también los productos que aparecen en pedidos del año 90. En este caso tenemos unos productos en la tabla de productos y los otros en la tabla de pedidos, las tablas no tienen las mismas columnas no se puede hacer una unión de ellas pero lo que interesa realmente es el identificador del producto (idfab,idproducto), luego por una parte sacamos los códigos de los productos con existencias cero (con una consulta), por otra parte los códigos de los productos que aparecen en pedidos del año 90 (con otra consulta), y luego unimos estas dos tablas lógicas.

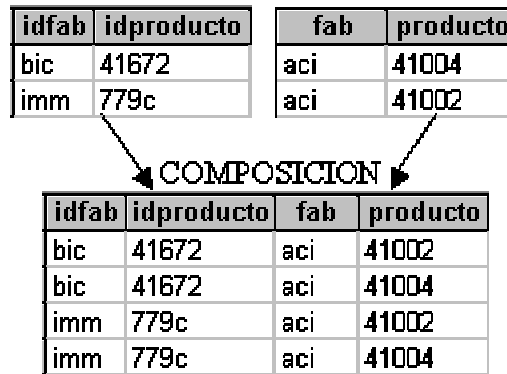
El operador que permite realizar esta operación es el operador **UNION**.



La composición de tablas

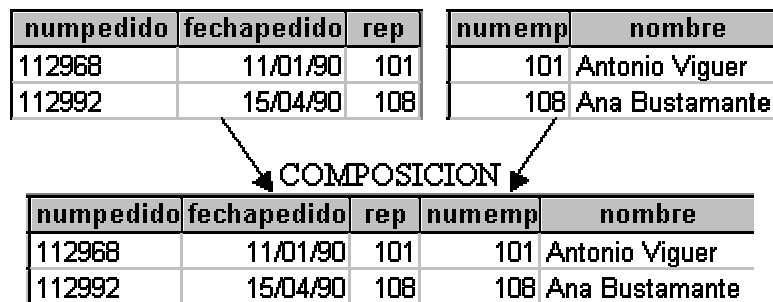
La composición de tablas consiste en **concatenar** filas de una tabla con filas de otra. En este caso obtenemos una tabla con las **columnas** de la **primera tabla unidas** a las **columnas** de la **segunda tabla**, y las filas de la tabla resultante son **concatenaciones** de **filas** de la **primera tabla con** filas de la **segunda tabla**.

El ejemplo anterior quedaría de la siguiente forma con la composición:



A diferencia de la unión la composición permite obtener una fila con datos de las dos tablas, esto es muy útil cuando queremos visualizar filas cuyos datos se encuentran en dos tablas.

Por ejemplo queremos listar los pedidos con el nombre del representante que ha hecho el pedido, pues los datos del pedido los tenemos en la tabla de pedidos pero el nombre del representante está en la tabla de empleados y además queremos que aparezcan en la misma línea; en este caso necesitamos componer las dos tablas (Nota: en el ejemplo expuesto a continuación, hemos seleccionado las filas que nos interesan).



Existen distintos tipos de composición, aprenderemos a utilizarlos todos y a elegir el tipo más apropiado a cada caso.

Los **tipos de composición** de tablas son:

El **producto cartesiano**

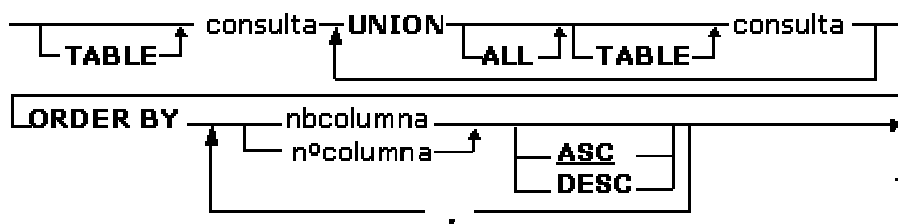
El **INNER JOIN**

El **LEFT / RIGHT JOIN**

El operador UNION

Como ya hemos visto en la página anterior, el operador **UNION** sirve para obtener a partir de dos **tablas** con las **mismas columnas**, una nueva tabla con las **filas** de la **primera** y las **filas** de la **segunda**.

La sintaxis es la siguiente:



Consulta puede ser un **nombre de tabla**, un **nombre de consulta** (en estos dos casos el nombre debe estar precedido de la palabra **TABLE**), o una **sentencia SELECT** completa (en este caso no se puede poner **TABLE**). La sentencia **SELECT** puede ser cualquier sentencia **SELECT** con la única restricción de que no puede contener la cláusula **ORDER BY**.

Después de la primera consulta viene la palabra **UNION** y a continuación la segunda consulta. La segunda consulta sigue las mismas reglas que la primera consulta.

Las dos consultas deben tener el **mismo número de columnas** pero las columnas pueden llamarse de **diferente** forma y ser de **tipos de datos distintos**.

Las **columnas del resultado** se llaman como las de la **primera consulta**.

Por **defecto** la unión **no incluye filas repetidas**, si alguna fila está en las dos tablas, sólo aparece una vez en el resultado.

Si **queremos** que aparezcan todas las filas incluso las **repeticiones de filas**, incluimos la palabra **ALL** (*todo* en inglés).

El empleo de **ALL** tienen una **ventaja**, la consulta **se ejecutará más rápidamente**. Puede que la diferencia no se note con tablas pequeñas, pero si tenemos tablas con muchos registros (filas) la diferencia puede ser notable.

Se puede **unir más de dos** tablas, para ello después de la segunda consulta **repetimos** la palabra **UNION** ... y así sucesivamente.

También podemos indicar que queremos el **resultado ordenado** por algún criterio, en este caso se incluye la cláusula **ORDER BY** que ya vimos en el tema anterior. La cláusula **ORDER BY** se escribe **después** de la **última consulta**, al final de la sentencia; para indicar las **columnas de ordenación** podemos utilizar su **número de orden** o el **nombre de la columna**, en este último caso se deben de utilizar los nombres de columna de la **primera consulta** ya que son los que se van a utilizar para nombrar las columnas del resultado.

Para ilustrar la operación vamos a realizar el ejercicio visto en la página anterior, vamos a obtener los códigos de los productos que tienen existencias iguales a cero o que aparezcan en pedidos del año 90.

```
SELECT idfab,idproducto
FROM productos
WHERE existencias = 0
UNION ALL
SELECT fab,producto
FROM pedidos
WHERE year(fechapedido) = 1990
ORDER BY idproducto
```

o bien

```
TABLE [existencias cero]
UNION ALL
TABLE [pedidos 90]
ORDER BY idproducto
```

Se ha incluido la cláusula **ALL** porque no nos importa que salgan filas repetidas.

Se ha incluido **ORDER BY** para que el resultado salga ordenado por idproducto, observar que hemos utilizado el nombre de la columna de la primera **SELECT**, también podíamos haber puesto **ORDER BY 2** pero no **ORDER BY producto** (es el nombre de la columna de la segunda tabla).

Para el 2º caso hemos creado una consulta llamada *existencias cero* con la primera **SELECT**, y una consulta llamada *pedidos 90* con la segunda **SELECT**. Observar que los nombres de las consultas están entre corchetes porque contienen espacios en blanco, y que en este caso hay que utilizar **TABLE**.

Estas serian en principio las consultas mas comunes a realizar en este lenguaje de consulta estructurado de bases de datos SQL. Por supuesto que podriamos extendernos mas aun si cabe incluyendo otro tipo de consultas que no se han visto en esta breve introduccion, como son, las consultas de actualizacion de una o mas lineas, las consultas de eliminacion, las consultas de creacion de tabla, las cuales ya se han podido ver directamente en el estudio del curso de access.