Vistas en SQL*

Miguel-Angel Sicilia

Abstract

Se describe el concepto de vista en esquemas relacionales y se introduce su sintaxis mediante ejemplos.

1 Vistas en SQL

Muchas bases de datos relacionales que se utilizan en aplicaciones del mundo real tienen esquemas complejos y formados por muchas tablas. En ocasiones, es conveniente que algunos grupos o perfiles de usuarios tengan una vista parcial de ese esquema, o que tengan una visión de la misma con una estructura diferente a la del esquema que realmente está almacenado. Precisamente para estos casos, el lenguaje SQL permite definir vistas.

Una vista es esencialmente una consulta almacenada que devuelve un conjunto de resultados y a la que se le pone un nombre. Una vista es una "tabla virtual", aparece como una tabla más del esquema, aunque realmente no lo es.

1.1 Sintaxis

La sintaxis general para crear una vista es la siguiente:

```
CREATE VIEW view_name [(column_list)]
AS sentencia_select
```

La idea es muy simple, solamente le damos nombre (`view_name`) a una consulta. Opcionalmente, los atributos de la relación resultante de la `sentencia_select` pueden renombrarse mediante etiquetas en `column_list`.

1.2 Un ejemplo

Tomemos como ejemplo una aplicación muy simple de gestión de pedidos en un supermercado virtual. El esquema relacional sería el siguiente.

*Version 1.1: Dec 8, 2008 5:10 pm US/Central
†http://creativecommons.org/licenses/by/2.0/

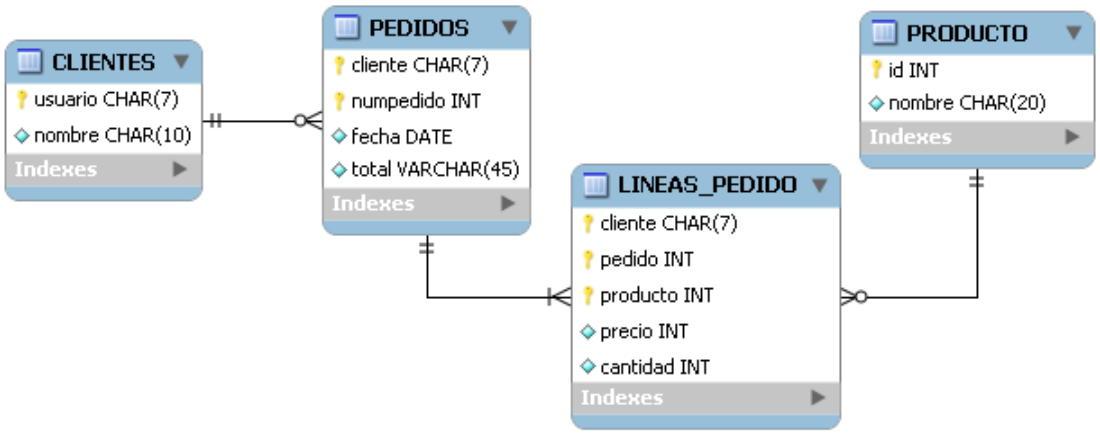

Figure 1

En ese esquema, la información aparece descompuesta en tablas. Sin embargo, para un usuario en un departamento de marketing, podría ser que le fuese más útil tener la información de las ventas de los productos acumuladas, simplemente.

```
CREATE VIEW resumenproductos AS
select p.id, p.nombre, sum(cantidad) AS total
from producto as p, lineas_pedido as l
where (l.producto = p.id)
group by l.producto order by total desc
```

Después de definir la vista, podremos utilizar resumenproductos como si fuese una tabla más. Por ejemplo la sentencia:

```
select * from resumenproductos
```

Nos devolverá el resultado de la consulta que define la vista.

Como segundo ejemplo, puede que una persona en Contabilidad solamente necesite el resumen económico de los pedidos. En ese caso, podremos definir una vista como la siguiente utilizando una subconsulta correlacionada:

```
CREATE VIEW resumenpedidos (usuario, nombre, pedido, fecha, total) ASSELECT c.usuario, c.nombre,
p.numpedido, p.fecha, (SELECT SUM(precio*cantidad) FROM LINEAS_PEDIDO as x WHERE (x.cliente
= p.cliente) and (x.pedido=p.numpedido))FROM CLIENTES as C, PEDIDOS as PWHERE p.cliente = c.usuario
```

El resultado sería como el siguiente:

usuario	nombre	pedido	fecha	total
agarcia	Ana Garcia	1	2008-11-05	40
jlopez	Juan Lopez	1	2008-02-10	65
jlopez	Juan Lopez	2	2008-02-11	null

Table 1

Nótese que cuando un pedido no tiene líneas asociadas, aparecerá un nulo en la subconsulta. La consulta puede hacerse también mediante agrupamiento con la siguiente consulta:

```
SELECT c.usuario, c.nombre, p.numpedido, p.fecha, sum(cantidad*precio)
FROM clientes as c, pedidos as p, lineas_pedido as l
WHERE (c.usuario = p.cliente) and (p.cliente = l.cliente) and (p.numpedido=l.pedido)
GROUP BY p.cliente, p.numpedido
```

En este caso se evita la aparición de nulos, ya que la cláusula `GROUP BY` no creará un subgrupo en el caso de que no haya líneas de pedido.

Lógicamente, los beneficios de las vistas se obtienen al combinar su definición con el sistema de permisos del gestor de base de datos. Siguiendo el ejemplo, daríamos permiso a los usuarios en Marketing sobre la vista `resumenproductos`, y permisos a los usuarios de Contabilidad sobre `resumenpedidos`.

1.3 La representación de las vistas

Dado que las vistas aparecen como tablas, pueden aparecer en otras consultas. Es importante tener esto en cuenta cuando se están diseñando consultas, dado que puede afectar al rendimiento. Para ello, algunos gestores de bases de datos tienen sintaxis extendidas para controlar cómo se representan internamente las vistas.

Por ejemplo, MySQL tiene una cláusula `ALGORITHM` que puede acompañarse de tres valores: `MERGE`,`TEMPTABLE` o `UNDEFINED`, con el siguiente significado:

- Con `MERGE`, el texto de las sentencias que hagan referencia a una vista se fusiona con el texto de la definición de la vista, de modo que las partes de la definición de la vista reemplazan a las partes correspondientes de la sentencia.
- Con `TEMPTABLE`, los resultados de la vista se recuperan en una tabla temporal, que se usa después para ejecutar la consulta.

El uso de `TEMPTABLE` consume un espacio temporal adicional, pero puede tener un mejor rendimiento ya que después de hacer la copia de los datos en la tabla temporal, se usa ésta y se libera la tabla o tablas originales.