

Consultas a múltiples tablas

Tema 4

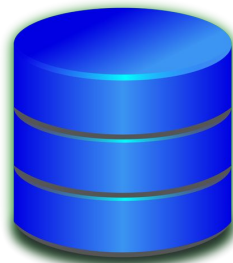
Bases de Datos - Grado en Ingeniería Informática
Antonio Balderas Alberico

Comentarios

- Transparencias correspondientes al **capítulo 5** del documento “Apuntes de prácticas” del Campus Virtual:
 - <https://av03-21-22.uca.es/moodle/mod/resource/view.php?id=72860>
- Para el lenguaje MySQL, se aportan referencias web oficiales donde se amplía y especifica el uso de las diferentes instrucciones.

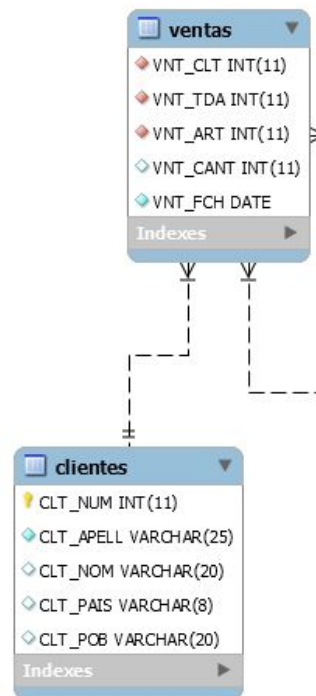
Introducción

- Las consultas suelen requerir información que está contenida en varias tablas → **se necesita que una columna contenga la misma información en diferentes tablas y este campo nos sirva de unión: **clave foránea****
- Operación: **producto natural** (join)



Producto natural

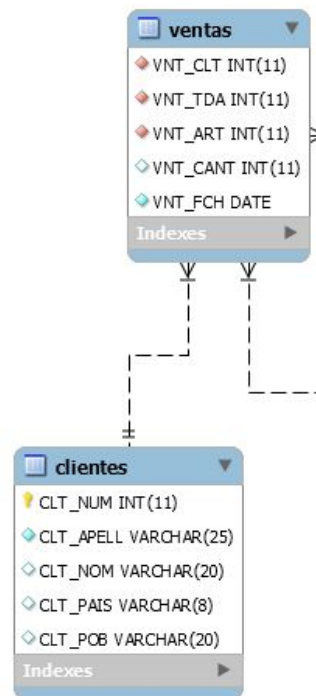
- ¿Qué clientes compraron después del 21 de febrero de 2020?



Producto natural

- ¿Qué clientes compraron después del 21 de febrero de 2020?

```
SELECT clt_num, clt_apell, clt_nom, vnt_fch
FROM clientes, ventas
WHERE clt_num = vnt_clt
AND vnt_fch > '20200221';
```

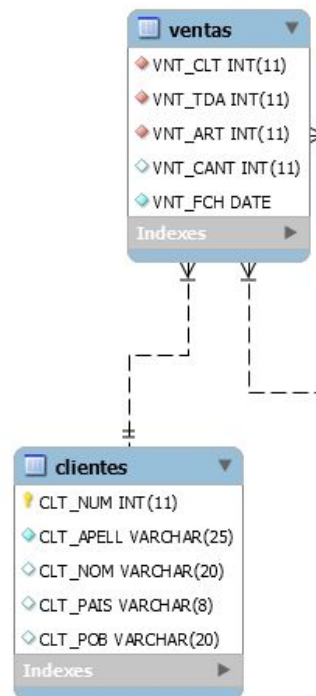


Producto natural

- ¿Qué clientes compraron después del 21 de febrero de 2020?

```
SELECT clt_num, clt_apell, clt_nom, vnt_fch
FROM clientes, ventas
WHERE clt_num = vnt_clt
AND vnt_fch > '20200221';
```

<u>clt_num</u>	<u>clt_apell</u>	<u>clt_nom</u>	<u>vnt_fch</u>
2	Perez	Miguel	2020-02-22
8	Courbon	Gerard	2020-02-29

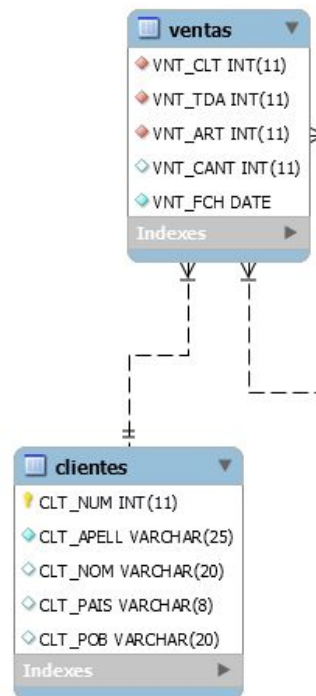


Producto natural

Características

- Si no ponemos la condición de unión obtenemos un **producto cartesiano** → Error grave, pues la tabla resultante del producto cartesiano entre 2 o más tablas contiene **información que no es cierta**

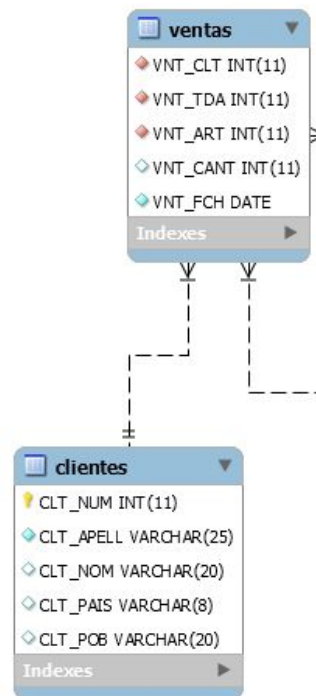
→ Más info del producto cartesiano en páginas 77 y 78 de los apuntes.



Producto natural

Características

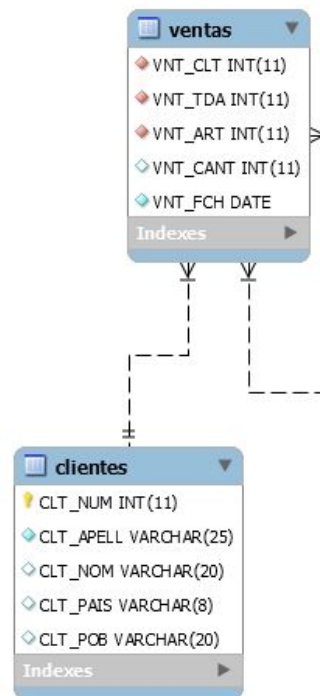
- Si no ponemos la condición de unión obtenemos un **producto cartesiano** → Error grave, pues la tabla resultante del producto cartesiano entre 2 o más tablas contiene **información que no es cierta**
- Puede usarse la expresión `nom_tabla.*` para ver todas las columnas de una tabla



Producto natural

Características

- Si no ponemos la condición de unión obtenemos un **producto cartesiano** → Error grave, pues la tabla resultante del producto cartesiano entre 2 o más tablas contiene **información que no es cierta**
- Puede usarse la expresión `nom_tabla.*` para ver todas las columnas de una tabla
- Para realizar un producto natural entre **n tablas**, es necesario tener al menos **$n-1$ condiciones de unión** para evitar productos cartesianos

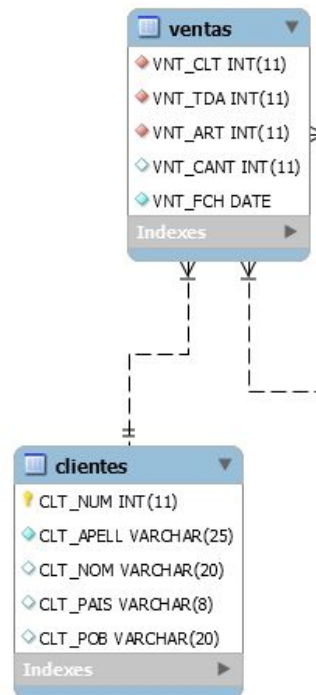


Producto natural y funciones de grupo

- Cantidad de compras realizadas por cada cliente

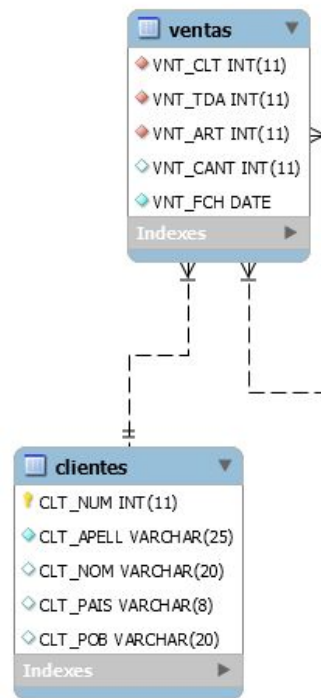
```
SELECT clt_num, clt_apell, clt_nom, count(*)  
FROM clientes, ventas  
WHERE clt_num = vnt_clt  
GROUP BY clt_num, clt_apell, clt_nom;
```

<u>clt_num</u>	<u>clt_apell</u>	<u>clt_nom</u>	<u>count(*)</u>
1	Borras	Margarita	4
2	Perez	Miguel	1



Autouniones

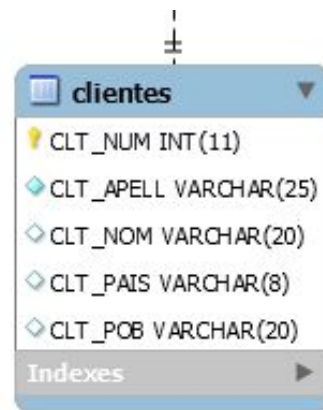
- Unión entre dos **tablas iguales**
- Una autounión une filas de una tabla con ella misma o con otras filas en la misma tabla
- El criterio de unión afecta al valor de una columna en relación al valor de esta misma columna en otra fila de la misma tabla
- Es necesario utilizar un **alias**



Autouniones

- Clientes que viven en la misma población de los que su número es superior a 10 y su nombre comienza por *m*

```
SELECT c.clt_num, c.clt_apell, c.clt_nom  
FROM clientes c, clientes d  
WHERE c.clt_pob = d.clt_pob  
AND d.clt_nom LIKE 'm%'  
AND d.clt_num > 10;
```



A screenshot of a database table structure for a table named 'clientes'. The table has five columns: CLT_NUM (INT(11)), CLT_APELL (VARCHAR(25)), CLT_NOM (VARCHAR(20)), CLT_PAIS (VARCHAR(8)), and CLT_POB (VARCHAR(20)). The 'Indexes' section is visible at the bottom with a right-pointing arrow.

clientes	
CLT_NUM	INT(11)
CLT_APELL	VARCHAR(25)
CLT_NOM	VARCHAR(20)
CLT_PAIS	VARCHAR(8)
CLT_POB	VARCHAR(20)
Indexes	

Unión externa

- Outer-joins
- Cuando se realiza una unión (producto natural), solamente aparecen aquellas filas que cumplen la condición de unión
- Gracias a la unión externa, podremos proyectar también aquellas filas que no cumplen el criterio de unión

Unión externa

Vamos a verlo con un ejemplo:

Listado de clientes y su última compra realizada:

Unión externa

Vamos a verlo con un ejemplo:

Listado de clientes y su última compra realizada:

```
SELECT clt_num, clt_nom, clt_apell, vnt_fch  
FROM clientes, ventas  
WHERE clt_num = vnt_clt  
GROUP BY clt_num  
ORDER BY clt_num;  
;
```

Unión externa

Vamos a verlo con un ejemplo:

Listado de clientes y su última compra realizada:

Misma consulta usando JOIN [1]

```
SELECT clt_num, clt_nom, clt_apell, vnt_fch  
FROM clientes JOIN ventas ON clt_num = vnt_clt  
GROUP BY clt_num  
ORDER BY clt_num;  
;
```


Unión externa

Vamos a verlo con un ejemplo:

Listado de clientes y su última compra realizada. También deberán aparecer aquellos clientes que aún no han comprado:

Misma consulta usando JOIN [2]

```
SELECT clt_num, clt_nom, clt_apell, vnt_fch  
FROM clientes LEFT OUTER JOIN ventas ON clt_num = vnt_clt  
GROUP BY clt_num  
ORDER BY clt_num;  
;
```

Operadores conjuntistas

- Unión
- Intersección
- Diferencia
- Producto cartesiano

Operadores conjuntistas

UNION

- Unión de los resultados de **dos consultas SELECT**
- Debe haber el **mismo número de columnas** en ambos SELECT y cada columna del primer SELECT debe ser del **mismo tipo** que la columna que ocupa su misma posición en el segundo SELECT
- No se muestran repeticiones a no ser que se incluya la cláusula **ALL**
- La cláusula ORDER BY actúa sobre la proyección resultado de la unión

Operadores conjuntistas

UNION

Ejemplo: listado de clientes que han comprado en enero junto con el listado de clientes que han comprado en febrero, ordenados por id.

Operadores conjuntistas

UNION

Ejemplo: listado de clientes que han comprado en enero junto con el listado de clientes que han comprado en febrero, ordenados por id.

```
SELECT * FROM clientes WHERE clt_num IN  
    (SELECT vnt_clt FROM ventas WHERE vnt_fch BETWEEN '2020-01-01' AND '2020-01-31')  
UNION [ALL]  
SELECT * FROM clientes WHERE clt_num IN  
    (SELECT vnt_clt FROM ventas WHERE vnt_fch BETWEEN '2020-02-01' AND '2020-02-29')  
ORDER BY clt_num
```

Operadores conjuntistas

UNION

Ejemplo: listado de clientes que han comprado en enero junto con el listado de clientes que han comprado en febrero, ordenados por id.

Esta consulta es equivalente a la siguiente:

```
SELECT * FROM clientes WHERE clt_num IN  
    (SELECT vnt_clt FROM ventas WHERE vnt_fch BETWEEN '2020-01-01' AND '2020-01-31')  
OR clt_num IN  
    (SELECT vnt_clt FROM ventas WHERE vnt_fch BETWEEN '2020-02-01' AND '2020-02-29')  
ORDER BY clt_num
```

Operadores conjuntistas

- Unión
- Intersección
- Diferencia
- Producto cartesiano

El resto de operadores no tienen un operador, como es el caso de la unión, pero sí se pueden implementar mediante los operadores que ya conocemos

Operadores conjuntistas

DIFERENCIA

Ejemplo: listado de clientes que han comprado en enero menos los clientes que han comprado en febrero, ordenados por id.

```
SELECT * FROM clientes WHERE clt_num IN  
    (SELECT vnt_clt FROM ventas WHERE vnt_fch BETWEEN '2020-01-01' AND '2020-01-31')  
AND clt_num NOT IN  
    (SELECT vnt_clt FROM ventas WHERE vnt_fch BETWEEN '2020-02-01' AND '2020-02-29')  
ORDER BY clt_num
```


Operadores conjuntistas

INTERSECCIÓN

Ejemplo: listado de clientes que han comprado en enero junto con el listado de clientes que han comprado en febrero, ordenados por id.

Esta consulta es equivalente a la siguiente:

```
SELECT * FROM clientes WHERE clt_num IN  
    (SELECT vnt_clt FROM ventas WHERE vnt_fch BETWEEN '2020-01-01' AND '2020-01-31')  
AND clt_num IN  
    (SELECT vnt_clt FROM ventas WHERE vnt_fch BETWEEN '2020-02-01' AND '2020-02-29')  
ORDER BY clt_num
```

Optimización de consultas

- Una consulta se suele poder expresar de distintos modos
- Cada una arroja tiempos de ejecución diferentes → depende del optimizador de cada SGBDR
- Ampliar información en la sección 5.4 del libro de prácticas

Optimización de consultas

- Una consulta se suele poder expresar de distintos modos
- Cada una arroja tiempos de ejecución diferentes → depende del optimizador de cada SGBDR
- Ampliar información en la sección 5.4 del libro de prácticas

Ejemplo: Listado de clientes que han comprado el artículo 3

→ Realice la consulta de 4 formas diferentes: unión entre tablas, consulta anidada, consulta correlacionada y consulta de existencia

Optimización de consultas

CONSEJOS

- Si la información a mostrar es de varias tablas: unión (producto natural)
- Si la información a mostrar es una única tabla: consulta anidada
- Toda subconsulta precedida por el operador IN puede expresarse con la ayuda del operador EXISTS (pero no al revés)
- EXISTS es necesario para expresar el cuantificador universal

Referencias

[1] MySQL: Join <https://dev.mysql.com/doc/refman/8.0/en/join.html>

[2] MySQL Outer Join: https://www.w3schools.com/sql/sql_join_left.asp

Bases de Datos (prácticas)

- Terminar de leer capítulo 5
- Ver vídeo presentación: https://youtu.be/xdyPSSyL_jU
- Hacer ejercicios del 11 al 20