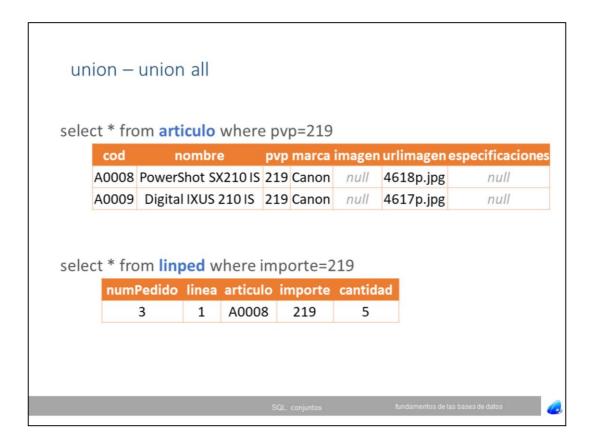
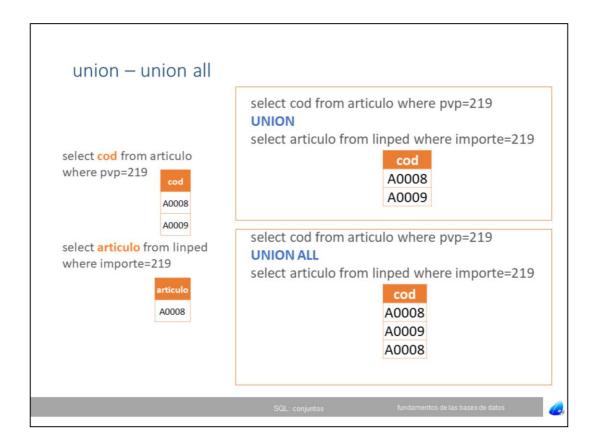




Operadores de conjuntos implementados y simulados o emulados



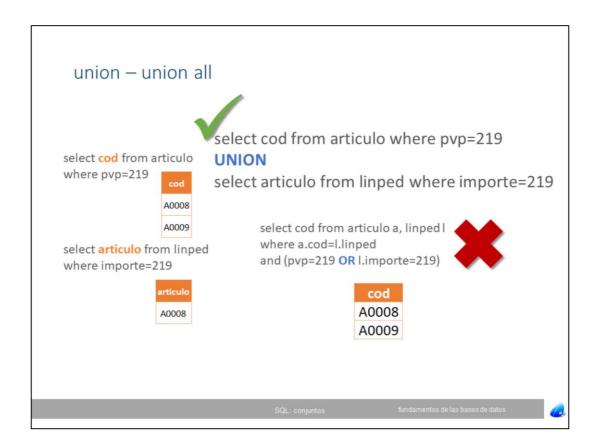
Dos consultas aparentemente independientes. Partimos de estos resultados.



Haciendo compatibles esas consultas —que devuelvan las mismas columnas y que sean del mismo tipo en las mismas posiciones— se puede operar con UNION. El resultado son todas las filas de ambas tablas eliminando duplicados.

Si no nos interesa que se eliminen filas duplicadas podemos usar UNION ALL.

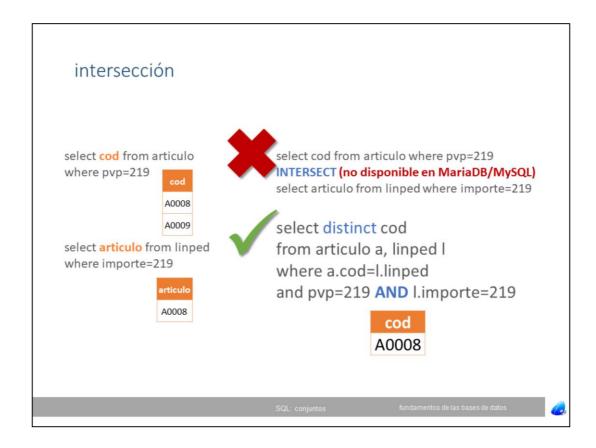
En muchos casos, es posible encontrar soluciones alternativas, aunque la complejidad de la expresión es algo mayor —incluso el plan de ejecución—.



Entre esas alternativas podemos pensar que una simple disyunción nos permite obtener el mismo resultado. Pero no, **obtenemos la misma tabla de salida de casualidad**.

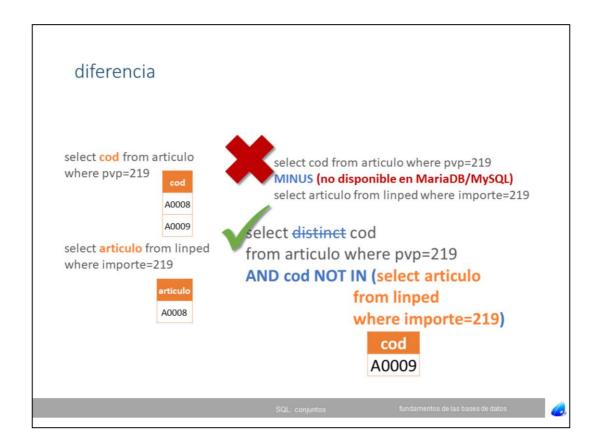
_	
union – union all	select cod from articulo where pvp=165 UNION select articulo from linped where importe=219
select cod from articulo where pvp=165 cod A0004 A0198	A0004 A0008 A0198
select articulo from linped where importe=219 articulo A0008	select cod from articulo a, linped I where a.cod=l.linped and (pvp=165 OR l.importe=219) cod A0008 A0198
	SQL: conjuntos fundamentos de las bases de datos.

En el ejemplo anterior **daba la casualidad** de que todos los códigos de artículo estaban en ambas tablas. Si no fuera ese el caso como, por ejemplo, el A0004 que solo se encuentra en ARTICULO, los resultados serían diferentes. Y **no podemos asegurar qué contienen las tablas** en cada momento.



Llevamos haciendo intersecciones inconscientemente desde casi que empezamos el curso. La concatenación de tablas (*join*), escrito de una manera o de otra, siempre obtiene intersecciones de conjuntos.

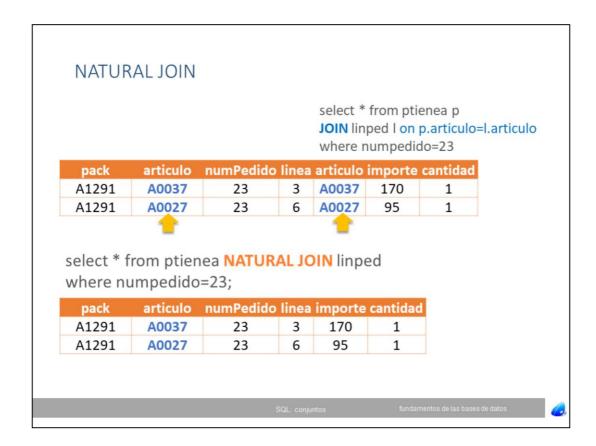
Utilizamos DISTINCT para ser fieles a la operativa de álgebra relacional. En este caso, además, es útil por sí solo, no parece que aporte mucho el obtener un montón de códigos de artículo repetidos.



La diferencia de conjuntos, ante relaciones compatibles, obtiene las tuplas que están en el primer conjunto pero no en el segundo. En MariaDB/MySQL no disponemos del operador MINUS, pero lo podemos sustituir por NOT IN.

En el ejemplo, códigos de artículo que no se encuentran entre los pedidos (la lista de códigos de artículo que se obtiene de LINPED).

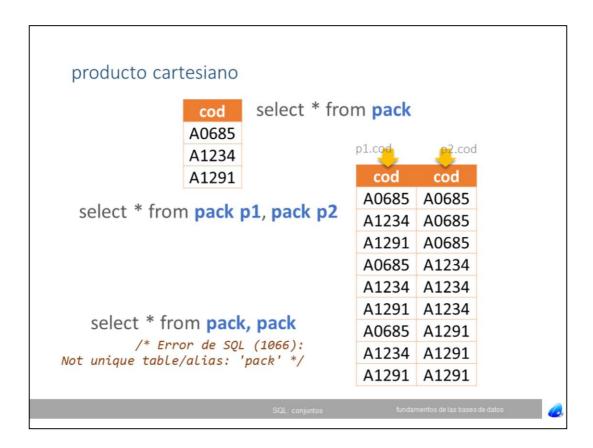
En este caso —y para ser fiel a los operadores del álgebra relacional que no producen tuplas duplicadas—, no hace falta utilizar DISTINCT porque pedimos ARTICULO.cod, que es clave primaria y no produce duplicados. En otros casos sí puede ser, como poco, recomendable.



NATURAL JOIN busca por nosotros columnas que tengan el mismo nombre y tipo de datos, y realiza la concatenación de filas. Además, elimina una de las columnas "comunes" ya que no es más que información redundante.

Al igual que siempre, debemos pensar que NATURAL JOIN se "ejecuta" antes que WHERE.

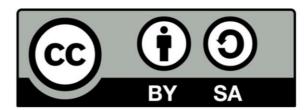
Recuerda que la primera consulta es totalmente equivalente a select * from ptienea p, linped l where p.articulo=l.articulo and numpedido=23.



El producto cartesiano es, simplemente, select... from... sin where que enlace las tablas.

Se puede poner incluso el producto de una tabla por sí misma, como en el ejemplo, siempre que especifiquemos un alias que distinga los nombres de las columnas en la tabla resultado.

licencias



Todos los logotipos y marcas registradas mostrados en este sitio son propiedad de sus respectivos propietarios y NO están bajo la licencia mencionada.

SQL: conjunto

fundamentos de las bases de datos