

## Consultas Multitaboa

---

## Índice

---

<b>1.</b>	<b>Consultas con datos de máis dunha táboa.....</b>	<b>3</b>
1.1	Consultas con datos de máis dunha táboa .....	3
1.2	Composición de táboas.....	3
	Nomes de columna cualificados .....	4
1.2.1	Composición interna.....	4
	Composición interna con INNER JOIN e condición de igualdade .....	5
	Composición interna con INNER JOIN sen condición de igualdade .....	6
1.2.2	Composición externa con OUTER JOIN .....	6
1.2.3	Composición con NATURAL JOIN.....	7
1.2.4	Composición dunha táboa con ela mesma. Autocomposición.....	8
	Normas para a realización de composicións internas e externas .....	9
1.3	Unión de consultas (UNION).....	9

# 1. Consultas con datos de máis dunha táboa

---

## 1.1 Consultas con datos de máis dunha táboa

A práctica diaria de consultas sobre unha base de datos implica frecuentemente a máis dunha táboa. Este tipo de consultas pódense realizar de varias formas en función do requirimento da consulta:

- Utilizar composicións de táboas nunha sentenza **SELECT** escribindo unha única consulta que utiliza a cláusula **JOIN** dentro da cláusula **FROM**.
- Utilizar sentenzas **SELECT** aniñadas dentro doutras sentenzas **SELECT**. As sentenzas aniñadas chámanse subconsultas.
- Unindo o conxunto de resultados dunha consulta co conxunto de resultados doutras consultas, empregando o operador **UNION**.

## 1.2 Composición de táboas



As primeiras normas ANSI SQL para combinar varias táboas nunha consulta permitían poñer a relación das táboas na cláusula **FROM** separadas por coma, e as condicións para facer os enlaces entre as táboas na cláusula **WHERE**. Esta sintaxe sigue estando permitida hoxe en día, aínda que non é a máis recomendable.

Exemplo: seleccionar apelidos, nome e cidade dos empregados que estean asignados a un departamento, tendo en conta que a cidade é a cidade na que está situado o departamento no que traballa o empregado.

```
/* Enlace entre dúas táboas utilizando unha clave foránea*/  
select apelidos, empregado.nome, cidade  
from empregado, departamento  
where departamento = codigo;
```

A primeira operación que se realiza cunha sentenza como a anterior na que se fai referencia a máis dunha táboa na cláusula **FROM**, é o produto cartesiano entre esas táboas, é dicir, relaciona cada fila dunha táboa con todas as filas da outra táboa. Por iso despois hai que poñer a condición de enlace na cláusula **WHERE** para que se seleccionen só aquelas composicións de filas que nos interesan.

O número de filas que devolve a consulta do exemplo é 19 tal e como se observa na imaxe seguinte, a pesar de que hai 20 empregados. Isto é debido a que hai un empregado que ten na columna *departamento* o valor **NULL**, por non estar asignado a ningún departamento, e non se pode combinar con ningunha fila da táboa *departamento* porque a columna *código* é clave primaria e non pode ter o valor **NULL**.

Result Grid				Filter Rows:
	apelidos	nome	cidade	
▶	Garcia Perez	Adrian	Ourense	
	Canedo Tellez	Angeles	Vigo	
	Nuñez Bernardes	Antonia	Monforte	
	Porto Novo	Begoña	Coruña	
	Martinez Iglesias	Benito	Lugo	
	Martinez Diaz	Carlos	Monforte	
	Ruiz Macias	Dario	Villalba	
	Abelleira Carrion	Dorinda	Villalba	

A sintaxe das composicións de táboas cambia a partir da norma ANSI92, para poñer as condicións de enlace na cláusula FROM xunto coa relación de táboas, coa finalidade de deixar a cláusula WHERE para as condicións que deben cumprir as filas que hai que mostrar.

Neste tema utilizamos a sintaxe que recomenda a norma ANSI92 por ser máis eficiente e estruturar mellor o código das consultas.

Resumo de tipo de composición máis utilizados		
Composición interna	INNER JOIN con condición de igualdade INNER JOIN sen condición de igualdade	
Composición externa	LEFT OUTER RIGHT OUTER	Prioridade esquerda Prioridade dereita
Composición natural	NATURAL JOIN	Pode ser interna ou externa
Autocomposición	Composición dunha táboa consigo mesma	Utilizando alias de táboas

## Nomes de columna cualificados

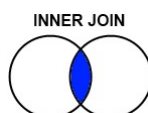
Nas combinacións de tipo interna, externa e autocomposición, hai que ter moi en conta que pode haber columnas que teñan o mesmo nome en dúas ou máis táboas. Cando sucede isto é obrigatorio cualificar os nomes das columnas, utilizando o formato *nome\_táboa.nome\_columna*, para evitar erros producidos polo uso de nomes ambiguos.

Tamén é recomendable utilizar os nomes de columna cualificados para mellorar o rendemento da consulta, porque proporcionan información a o servidor que desta maneira non ten que buscar a que táboa pertence cada columna. Para simplificar o uso de nomes cualificados e que estes sexan máis curtos, recoméndase utilizar alias para os nomes de táboa. Exemplo:

```
select em.apelidos, em.nome, de.cidade
from empregado as em, departamento as de
where em.departamento = de.codigo;
```

### 1.2.1 Composición interna

A composición interna crea unha táboa que contén todas as columnas das táboas que forman a composición e só as filas que cumpren as condicións da composición.



## Composición interna con INNER JOIN e condición de igualdade

Este tipo de composición é a máis utilizada e permite relacionar dúas táboas que teñen algunha columna con datos comúns que serve de enlace entre elas. O caso máis normal é o que establecen as claves foráneas dunha táboa, que toman valores que teñen que coincidir cos valores da clave primaria doutra táboa. Esta composición dá como resultado o conxunto formado polas parellas de filas das dúas táboas que interveñen na composición, que cumpran a condición de que o valor da clave foránea dunha sexa igual ao valor da clave primaria da outra. As columnas de enlace deben ter asociados índices para optimizar o rendemento da consulta.

A sintaxe utilizada na norma ANSI92 para a reunión interna é:

```
FROM nome_táboa_1 [ INNER ] JOIN nome_táboa_2
ON ( condición de enlace )
```

A opción INNER indica o tipo de composición interna, pero non é necesario poñela explicitamente. Cando se pon só a palabra JOIN entre os nomes de dúas táboas enténdese que é unha composición interna (INNER).

Exemplo: seleccionar *apelidos*, *nome* e *cidade* dos empregados que estean asignados a un departamento, tendo en conta que *cidade* é a cidade na que está situado o departamento no que traballa o empregado. A condición de enlace escríbese na cláusula FROM, reservando a cláusula WHERE para establecer condicións que seleccionen as filas das táboas de orixe que nos interesan.

```
/* Enlace entre dúas táboas utilizando unha clave foránea
con INNER JOIN segundo a norma ANSI-92*/
select em.apelidos, em.nome, de.cidade
from empregado as em inner join departamento as de on (em.departamento = de.codigo);
```

Cando interveñen máis de dúas táboas na composición, hai que ter en conta que para cada JOIN ten que existir unha condición de enlace. Neste caso, pódese asociar unha cláusula ON a cada JOIN coa correspondente condición de enlace, ou ben poñer unha única cláusula ON cunha condición composta formada por todas as condicións de enlace relacionadas con operadores AND.

Exemplo: mostrar *apelidos* e *nome* do empregado, nome do departamento no que traballa e nome da provincia no que está o departamento.

```
select em.apelidos, em.nome, de.nome, pr.provincia
from empregado as em
join departamento as de on (em.departamento = de.codigo)
join provincia as pr on (de.id_provincia = pr.id_provincia);
```

Tamén se podería escribir utilizando unha única condición de enlace composta:

```
select em.apelidos, em.nome, de.nome, pr.provincia
from empregado as em
join departamento as de
join provincia as pr
on (em.departamento = de.codigo and de.id_provincia = pr.id_provincia);
```

Calquera destas consultas obtén o seguinte resultado:

Result Grid				
		Filter Rows:		Exp
	apelidos	nome	nome	provincia
▶	Martinez Iglesias	Benito	Central	Lugo
	Fernandez Lopez	Jose Luis	Central	Lugo
	Fernandez Diaz	Julian	Central	Lugo
	Nuñez Bernardéz	Antonia	Oficina1	Lugo
	Martinez Diaz	Carlos	Oficina1	Lugo
	Hernandez Valin	Valentina	Oficina2	Coruña, A

## Composición interna con INNER JOIN sen condición de igualdade

Cando se fai unha composición entre dúas táboas utilizando unha composición interna, é posible utilizar condicións de enlace distintas da condición de igualdade; neste caso suponse que non existe ningunha columna que conteña valores idénticos nas dúas táboas.

Exemplo: seleccionar *apelidos*, *nome*, salario bruto, tipo de imposto e salario neto (salario bruto – impostos) de todos os empregados ordenados por nome.

```
/* JOIN sen condición de igualdade */
select em.apelidos, em.nome, em.salario_bruto, ir.tipo_imposto,
       round(em.salario_bruto-em.salario_bruto*ir.tipo_imposto/100,2) as salario_netos
from empregado as em join irpf as ir
     on em.salario_bruto between ir.limite_inferior and ir.limite_superior
order by em.nome;
```

Neste caso, o *salario\_bruto* é a columna da táboa *empregado* que se vai a utilizar como enlace e non ten que coincidir exactamente co valor almacenado en ningunha columna da táboa *irpf*, senón que ten que estar entre un *limite\_inferior* e un *limite\_superior*, ambos inclusive. O resultado da execución en Workbench podería ser:

Result Grid			Filter Rows:	Export: 	Wrap Cell Content: 
	apelidos	nome	salario_bruto	tipo_imposto	salario_netos
▶	Iglesias Dominguez	Adolfo	52500.00	27.00	38325.00
	Garcia Perez	Adrian	21500.00	21.00	16985.00
	Canedo Tellez	Angeles	58500.00	30.00	40950.00
	Nuñez Bernardez	Antonia	42000.00	27.00	30660.00
	Porto Novo	Begoña	52000.00	27.00	37960.00
	Martinez Iglesias	Benito	25000.00	21.00	19750.00

## 1.2.2 Composición externa con OUTER JOIN

A composición externa permite mostrar as filas dunha das táboas aínda que non cumpran a condición de enlace (toma o valor NULL para as columnas do resto de táboas), ademais das parellas de filas para as que se cumpra a condición de enlace tal e como ocorrería na composición interna.



Existen dous tipos de reunión externa: pola esquerda, e pola dereita, dependendo de cal das táboas se considere táboa principal. A sintaxe é:

```
FROM nome_táboa_1 { LEFT | RIGTH } [ OUTER ] JOIN nome_táboa_2
ON ( condición de enlace )
```

- A opción **LEFT** mostra todas as filas da táboa da esquerda, aínda que non estean relacionadas con filas do resto das táboas.
- A opción **RIGHT**, mostra todas as filas da táboa da dereita, aínda que non estean relacionadas con filas do resto das táboas.

Exemplo: mostrar *apelidos* e *nome* dos empregados e nome da cidade na que está o departamento no que traballa, aínda que o empregado non teña departamento asignado. O resultado ten que estar ordenado polo nome.

```

/* OUTER JOIN enlace externo, neste caso, pola esquerda*/
select em.apelidos, em.nome, de.cidade
from empregado as em left join departamento as de
on (em.departamento = de.codigo)
order by em.nome;

```

Na sentenza anterior, a táboa principal é a táboa *empregado* por estar situada á esquerda (*left*), e a táboa *departamento* considérase a táboa secundaria da consulta. Na execución, móstrase unha fila para cada empregado, e para os que non cumpran a condición de enlace por non ter asignado aínda un departamento, móstrase o valor NULL na columna *cidade*. O número de filas que devolve a consulta é 20 que é o número de filas que ten a táboa *empregado*.

Result Grid			Filter Rows:
	apelidos	nome	cidade
▶	Iglesias Dominguez	Adolfo	NULL
	Garcia Perez	Adrian	Ourense
	Canedo Tellez	Angeles	Vigo
	Nuñez Bernardez	Antonia	Monforte
	Porto Novo	Begoña	Coruña
	Martinez Iglesias	Benito	Lugo
	Martinez Diaz	Carlos	Monforte
	Ruiz Macias	Dario	Villalba

Exemplo de LEFT JOIN con IF NULL: mostrar *apelidos* e *nome* dos empregados que non teñen departamento asignado.

```

select em.apelidos, em.nome
from empregado as em left join departamento as de
on (em.departamento = de.codigo)
where de.cidade is null; # Vale calquera columna da táboa departamento

```

Para as filas da táboa principal (*empregado*) que non cumpran a condición de enlace, todas as columnas da táboa secundaria (*departamento*) toman o valor NULL, polo que ao establecer a condición da cláusula WHERE só se mostran as filas da táboa principal que non están relacionadas con ningunha fila da táboa secundaria.

Result Grid	Filter Rows:
apelidos	nome
Iglesias Dominguez	Adolfo

### 1.2.3 Composición con NATURAL JOIN

A composición natural permite enlazar táboas por columnas que teñen o mesmo nome. Neste tipo de composicións non é necesario introducir a condición de enlace coa opción ON, pois sobreenténdese que a condición de enlace consiste en comprobar a coincidencia dos valores das columnas que teñen o mesmo nome. Ademais, as columnas que existen nas dúas táboas só se mostran unha vez, polo que non é necesario cualificalas empregando o formato *nome\_táboa.nome columna*. Sintaxe:

```

FROM nome_táboa_1 NATURAL [{ LEFT | RIGTH } [ OUTER ]] JOIN nome_táboa_2

```

A combinación con NATURAL JOIN pode dar prioridade a algunha das táboas, cando se define como enlace externo (OUTER).

Cando se utiliza NATURAL JOIN hai que ter coidado que nas dúas táboas só teñan o mesmo nome as columnas que se van a utilizar para facer o enlace, xa que en calquera ou-

tro caso os resultados obtidos poden ser imprevisibles.

Exemplo: mostrar os datos de todas as columnas da táboa *departamento*, completados co nome da provincia da táboa *provincia*.



```
/* Utilización de NATURAL JOIN */
select de.*, pr.provincia
from departamento as de natural join provincia as pr
order by de.codigo;
```

Result Grid		Filter Rows:			Export:
	codigo	nome	cidade	id_provincia	provincia
▶	1	Central	Lugo	27	Lugo
	2	Oficina1	Monforte	27	Lugo
	3	Oficina2	Ferrol	15	Coruña, A
	4	Oficina3	Vigo	36	Pontevedra
	5	Oficina4	Ourense	32	Ourense

Na mesma consulta pódense utilizar distintos tipos de composición.

Exemplo: mostrar apelidos e nome do empregado, nome do departamento no que traballa e nome da provincia no que está o departamento.

```
/* Utilización de dúas composicións, una delas NATURAL JOIN */
select em.apelidos, em.nome, de.nome, pr.provincia
from empregado as em join departamento as de on (em.departamento = de.codigo)
natural join provincia as pr;
```

Result Grid			 Filter Rows:	E
	apelidos	nome	nome	provincia
▶	Martinez Iglesias	Benito	Central	Lugo
	Fernandez Lopez	Jose Luis	Central	Lugo
	Fernandez Diaz	Julian	Central	Lugo
	Nuñez Bernardes	Antonia	Oficina 1	Lugo
	Martinez Diaz	Carlos	Oficina 1	Lugo

## 1.2.4 Composición dunha táboa con ela mesma. Autocomposición

A autocomposición utilízase cando hai que comparar unha fila dunha táboa con outras filas da mesma táboa. Para evitar que ao ler unha fila da táboa se perda a información da anterior fila lida, pódese poñer dúas veces o nome da táboa na cláusula FROM, asignándolle dous alias ou sinónimos diferentes; a partir dese momento, manéxanse como se foran dúas táboas distintas. As consultas feitas cunha autocomposición poden ser resoltas tamén empregando subsentenzas.

Exemplo: mostrar apelidos e nome dos empregados, e apelidos e nome do seu xefe.

```
/* Autocomposición: Enlace dunha táboa con ela mesma */
select em1.apelidos, em1.nome, em1.departamento,
       em2.apelidos as Apelido_xefe, em2.nome as Nome_xefe
from empregado as em1 join empregado as em2
on em1.dni_xefe = em2.dni
order by Apelido_xefe, Nome_xefe;
```



Result Grid		Filter Rows:	Export:	Wrap Cell Content:
apellidos	nome	departamento	apellido_xefe	nome_xefe
Hernandez Valin	Valentina	3	Bernardez Macia	Luisa
Quiroga Juarez	Francisco	3	Bernardez Macia	Luisa
Sanchez Rodriguez	Maria	3	Bernardez Macia	Luisa
Vila Bernal	Rosario	4	Canedo Tellez	Angeles
Aguiar Lopez	Luis	4	Canedo Tellez	Angeles
Case Rodriguez	Fernanda	4	Canedo Tellez	Angeles
Garcia Perez	Adrian	5	Cendan Villa	Lorenzo

## Normas para a realización de composicións internas e externas

Non existen normas estándar para as composicións internas e externas, pero a continuación enuméranse unha serie de conclusións sobre as composicións internas e externas que axudarán a utilizalas mellor.

- As columnas que se utilizan para facer a composición deben ser do mesmo tipo. Para mellorar o rendemento da consulta deben ter asociados índices.
- Cando se fai unha composición, é como se crease unha táboa que ten as columnas de todas as táboas que se combinan, e as filas que verifican as condicións de composición. Por esta razón, na sentenza **SELECT** pódese utilizar calquera columna das táboas que se relacionan na cláusula **FROM**.
- Pódense combinar tantas táboas como desexemos, pero poñer táboas innecesarias reduce o rendemento da consulta.
- Unha mesma táboa pode aparecer máis dunha vez na composición, pero utilizando alias diferentes.
- Recoméndase utilizar nomes de columna cualificados. Ver o apartado '*Nomes de columna cualificados*' ao principio do documento.
- Se as condicións de composición van na cláusula **WHERE**, hai que ter en conta que primeiro debemos poñer todas aquelas condicións que non sexan de composición. Deste xeito o produto cartesiano previo ao *join* terá menos filas que combinar.
- Se unha táboa ten unha clave primaria composta, nas condicións de composición hai que facer referencia á clave enteira. Exemplo: Temos unha táboa *t1* que ten unha clave primaria composta formada polas columnas (*cp1, cp2*), e unha táboa *t2* que ten unha clave foránea composta polas columnas (*cf1, cf2*) que fai referencia á clave primaria da táboa *t1*. A condición de composición podería ser:

$t1.cp1 = t2.cf1 \text{ and } t1.cp2 = t2.cf2$

Condición composta

$(t1.cp1, t1.cp2) = (t2.cf1, t2.cf2)$

Comparación de valores tipo fila

## 1.3 Unión de consultas (UNION)

A cláusula **UNION** utilízase para combinar o resultado de varias consultas nun único conxunto de resultados. A sintaxe é:

```
Sentenza SELECT
UNION [ALL | DISTINCT]
Sentenza SELECT
[UNION [ALL | DISTINCT]
Sentenza SELECT] ...
```

- As opcións ALL e DISTINCT permiten indicar se hai que mostrar as filas duplicadas ou non. O valor por defecto é DISTINCT. Para que se mostren as filas duplicadas hai que poñer a opción ALL.
- As sentenzas SELECT que se utilizan na UNION teñen que ter o mesmo número de columnas na lista de selección. As columnas seleccionadas teñen que ser do mesmo tipo en todas as sentenzas SELECT. Por exemplo, se a primeira consulta selecciona tres columnas, o resto das sentenzas SELECT deben seleccionar tamén tres columnas; se a primeira columna da primeira consulta é de tipo char(50), a primeira columna do resto de sentenzas SELECT ten que ser de tipo char(50).
- Os nomes das columnas da táboa resultante da UNION son os que corresponden aos nomes das columnas da primeira sentenza SELECT.
- Restricións para as sentenzas SELECT que participan nunha unión:
  - A cláusula INTO OUTFILE só a pode levar a última sentenza.
  - Non se pode utilizar a opción HIGH\_PRIORITY. Non se mostra erro no caso de poñela na primeira sentenza, pero non se ten en conta. No caso de poñela en calquera das outras sentenzas, móstrase unha mensaxe de erro de sintaxe.
- Cando se queren utilizar as cláusulas ORDER BY ou LIMIT, hai que pechar as sentenzas SELECT entre parénteses.
- Non ten efecto a cláusula ORDER BY dentro dos parénteses, xa que a UNION produce un conxunto de filas desordenadas. Só ten sentido utilizar a cláusula ORDER BY para unha sentenza SELECT individual cando se combina coa cláusula LIMIT.

Exemplo extraído do manual de referencia de MySQL 5.6:

```
(SELECT a FROM tbl_name WHERE a=10 AND B=1 ORDER BY a LIMIT 10)
UNION
(SELECT a FROM tbl_name WHERE a=11 AND B=2 ORDER BY a LIMIT 10);
```

- No caso de querer ordenar as filas que forman o conxunto de resultados, hai que poñer a cláusula ORDER BY fóra dos parénteses, despois da última sentenza SELECT. Tamén é posible limitar o número de filas que forman o conxunto de resultados poñendo a cláusula LIMIT fóra dos parénteses, despois da última sentenza SELECT.

Exemplo extraído do manual de referencia de MySQL 5.6:

```
(SELECT a FROM tbl_name WHERE a=10 AND B=1)
UNION
(SELECT a FROM tbl_name WHERE a=11 AND B=2)
ORDER BY a LIMIT 10;
```