Lecciones SQL >

T05 Reunión (join)

El uso de varias tablas en una consulta, y su concatenación siguiendo cualquier criterio, se conoce habitualmente como *join*, el término en inglés adoptado y utilizado. Estamos hablando, por ejemplo, de:

```
select *
from asignaturas, profesores, imparte
where profesores.dni = imparte.dni and asignatura =
codigo
```

Contenidos

Tipos de reunión

Dependiendo de qué tablas se especifiquen en el *from* y del tipo de condición exigida para relacionar las filas de esas tablas, el *join* recibe distintos nombres (terminología que se puede considerar estándar):

Equijoin

Consultas que conllevan el uso de igualdades para la concatenación de filas de varias tablas. El ejemplo anterior es una *equijoin*.

Self join

Estas consultas concatenan una tabla consigo misma:

```
select i1.dni, ' imparte la misma asignatura que ', i2.dni
from imparte i1, imparte i2
where i1.asignatura= i2.asignatura
```

dni	imparte la misma asignatura que	dni
21111222	imparte la misma asignatura que	21111222
21111222	imparte la misma asignatura que	21111222
21333444	imparte la misma asignatura que	21333444

Producto cartesiano

El producto cartesiano utiliza dos tablas sin la condición de concatenación del *where*:

```
select dni, codigo
from profesores, asignaturas
```

dni	codigo

21111222	DGBD
21222333	DGBD
21333444	DGBD
21111222	FBD
21222333	FBD
21333444	FBD
21111222	FP
21222333	FP
21333444	FP
21111222	HI
21222333	HI
21333444	HI
21111222	PC
21222333	PC
21333444	PC

Inner join, simple join

Realmente, se trata de la consulta habitual en SQL. Sin embargo, existe una sintaxis particular alternativa usando *inner join*:

```
select nombre, descripcion
from asignaturas
     join imparte on (codigo=asignatura)
     join profesores on (imparte.dni=profesores.dni)
```

nombre	descripcion
EVA GOMEZ	DISEÑO Y GESTION DE BASES DE DATOS
EVA GOMEZ	FUNDAMENTOS DE LAS BASES DE DATOS
RAFAEL ROMERO	PROGRAMACION CONCURRENTE

El resultado será la concatenación de todas aquellas filas, y únicamente esas, que cumplen la condición que las relaciona. Es una construcción alternativa a la que hemos venido utilizando hasta ahora que nos debe ser familiar:

```
select nombre, descripcion
from asignaturas, profesores, imparte
where profesores.dni = imparte.dni and asignatura = codigo
```

Si acaso, puede tener cierta utilidad para no olvidarnos de enlazar cada par de tablas y dejar el *where* para otro tipo de condiciones:

```
select nombre, descripcion
from asignaturas join imparte on (codigo=asignatura)
join profesores on (imparte.dni=profesores.dni)
where descripcion NOT LIKE 'PROGRAMACION%'
```

nombre	descripcion
EVA GOMEZ	DISEÑO Y GESTION DE BASES DE DATOS
EVA GOMEZ	FUNDAMENTOS DE LAS BASES DE DATOS

Outer join

El outer join se diferencia del inner join en que las filas de una tabla que se muestran en el resultado no necesariamente tienen su correspondiente fila o filas en la otra tabla.

Por ejemplo, podríamos querer obtener todos los profesores y, si da alguna asignatura, el código de esas asignatura:

```
select p.*, i.asignatura
from profesores p
left join imparte i on (p.dni=i.dni);
```

dni	nombre	categoria	ingreso	asignatura
21111222	EVA GOMEZ	TEU	1993-10-01	DGBD
21111222	EVA GOMEZ	TEU	1993-10-01	FBD
21222333	MANUEL PALOMAR	TEU	1989-06-16	
21333444	RAFAEL ROMERO	ASO6	1992-06-16	PC

Más adelante se profundiza en el *outer join*.

Antijoin

Estas consultas son las que utilizan el NOT IN para obtener aquellas filas de una tabla que no se relacionan con las de otra:

```
select * from profesores
where dni not in (select dni from imparte);
```

dni	nombre	categoria	ingreso
21222333	MANUEL PALOMAR	TEU	1989-06-16

Semijoin

Un *semijoin* devuelve filas que hacen cierta una subconsulta de un operador EXISTS sin duplicar filas.

```
select * from profesores p
where exists (select * from imparte i where i.dni = p.dni);
```

dni	nombre	categoria	ingreso
21111222	EVA GOMEZ	TEU	1993-10-01
21333444	RAFAEL ROMERO	ASO6	1992-06-16

El operador *exists* y este tipo de subconsultas se verán en sesiones más avanzadas.

Démonos cuenta que, prácticamente, lo único que hemos hecho es dar nombre a los distintos tipos de consultas, algunas ya utilizadas durante el curso. No es importante ese nombre sino entender las necesidades de cada consulta y cómo satisfacerla.

Sí es nueva, para este curso, la sintaxis del *inner* y el *outer join*. El primero, no hace falta desarrollarlo más, el segundo sí lo tratamos en la siguiente sección.

Téngase en cuenta, también, que éste no es un curso exhaustivo de SQL. Hay detalles de rendimiento que favorecen el uso de unos u otros tipos de consultas y, por supuesto, muchas más opciones a la hora de incrementar ese rendimiento sobre todo en entornos de medianas o grandes bases de datos.

Para más información, podéis consultar Oracle® Database SQL Reference 10g Release 2 (10.2), que ha sido la fuente de esta sesión.

Outer join

Como ya hemos dicho, el *outer join* extiende el resultado de una consulta simple de, por ejemplo, dos tablas, obteniendo todas las filas que cumplen la condición de concatenación y, además, todas o algunas de las filas de una tabla que no cumplen tal condición.

Supongamos dos tablas A y B:

- `select * from A left [outer] join B on (condición)`
  Obtiene todas las filas relacionadas de A y B, y todas las no relacionadas de A.

- `select * from A right [outer] join B on (condición)`
  Obtiene todas las filas relacionadas de A y B, y todas las no relacionadas de B.

- `select * from A full [outer] join B on (condición)`
  (No soportado por MySQL) Obtiene todas las filas relacionadas de A y B, y todas las no relacionadas de A y B.

> **BD Ejemplo**

PROFESORES (**dni** varchar2(10), **nombre** varchar(40), **categoria** char(4), **ingreso** date)
CP (dni)

ASIGNATURAS (**codigo** char(5), **descripcion** varchar(35), **creditos** decimal(3,1), **creditosp** decimal(3,1))
CP (código)

IMPARTE (**dni** varchar(10), **asignatura** char(5))
CP (dni, asignatura)
CAj (dni) → PROFESORES
CAj (asignatura) → ASIGNATURAS

COORDINADORES (**dni** varchar(10), **nombre** varchar(40), **dpto** char(4), **asig** char(5))
CP (dni)
CAj (asig) → ASIGNATURAS

Para ver mejor el funcionamiento de las distintas alternativas de *join*, vamos a trabajar con una tabla adicional, COORDINADORES, en nuestra base de datos Ejemplo.

Muestra todos los coordinadores y, si lo hacen, las asignaturas que coordinan.

```
select * from coordinadores left join asignaturas on (codigo=asig);
```

dni	nombre	dpto	asig	codigo	descripcion	creditos	creditosp
55777666	AGAPITO CIFUENTES	DLSI	FP	FP	FUNDAMENTOS DE LA PROGRAMACION	9.0	4.5
66555444	ROMUALDO GOMEZ	DLSI	HI	HI	HISTORIA DE LA INFORMATICA	4.5	
99222111	CATURLO PEREZ	DLSI					

Muestra los coordinadores que tienen asignatura y todas las asignaturas.

```
select * from coordinadores right join asignaturas on (codigo=asig);
```

dni	nombre	dpto	asig	codigo	descripcion	creditos	creditosp
				DGBD	DISEÑO Y GESTION DE BASES DE DATOS	6.0	3.0
				FBD	FUNDAMENTOS DE LAS BASES DE DATOS	6.0	1.5
55777666	AGAPITO CIFUENTES	DLSI	FP	FP	FUNDAMENTOS DE LA PROGRAMACION	9.0	4.5
66555444	ROMUALDO GOMEZ	DLSI	HI	HI	HISTORIA DE LA INFORMATICA	4.5	
				PC	PROGRAMACION CONCURRENTE	6.0	1.5

Muestra todos los coordinadores y todas las asignaturas y si hay relación entre ellos.

```
select * from coordinadores full join asignaturas on (codigo=asig);
```

Lo que se espera de un *full join* es que aparezcan todos los datos de una y otra tabla, estén o no relacionados, más o menos, lo que se muestra a continuación:

dni	nombre	dpto	asig	codigo	descripcion	creditos	creditosp
				DGBD	DISEÑO Y GESTION DE BASES DE DATOS	6.0	3.0
55777666	AGAPITO CIFUENTES	DLSI	FP	FP	FUNDAMENTOS DE LA PROGRAMACION	9.0	4.5
66555444	ROMUALDO GOMEZ	DLSI	HI	HI	HISTORIA DE LA INFORMATICA	4.5	
				FBD	FUNDAMENTOS DE LAS BASES DE DATOS	6.0	1.5
				PC	PROGRAMACION CONCURRENTE	6.0	1.5
99222111	CATURLO PEREZ	DLSI					

Sin embargo, *full join* no está soportado por MySQL aunque sí por otros motores (Oracle PL/SQL) y si ejecutáramos la sentencia anterior el resultado es idéntico a un *join* simple.

dni	nombre	dpto	asig	codigo	descripcion	creditos	creditosp
55777666	AGAPITO CIFUENTES	DLSI	FP	FP	FUNDAMENTOS DE LA PROGRAMACION	9.0	4.5
66555444	ROMUALDO GOMEZ	DLSI	HI	HI	HISTORIA DE LA INFORMATICA	4.5	