Unidad Didáctica 7. Consultas sobre varias tablas. Composición externa

JOSÉ JUAN SÁNCHEZ HERNÁNDEZ

IES Celia Viñas (Almería) - 2020/2021

Índice general

Índice de cuadros

Índice de figuras

Capítulo 1

Consultas sobre varias tablas. Composición externa

1.0.1 Composiciones externas

- Join externa
  - `LEFT OUTER JOIN`
  - `RIGHT OUTER JOIN`
  - `FULL OUTER JOIN` (No implementada en MySQL)
  - `NATURAL LEFT OUTER JOIN`
  - `NATURAL RIGHT OUTER JOIN`

Ejemplo de `LEFT OUTER JOIN`:

```
SELECT *
FROM empleado LEFT JOIN departamento
ON empleado.codigo_departamento = departamento.codigo
```

Esta consulta devolverá todas las filas de la tabla que hemos colocado a la izquierda de la composición, en este caso la tabla `empleado`. Y relacionará las filas de la tabla de la izquierda (`empleado`) con las filas de la tabla de la derecha (`departamento`) con las que encuentre una coincidencia. Si no encuentra ninguna coincidencia, se mostrarán los valores de la fila de la tabla izquierda (`empleado`) y en los valores de la tabla derecha (`departamento`) donde no ha encontrado una coincidencia mostrará el valor `NULL`.

LEFT JOIN

1

```
/* SQL 2 */
SELECT *
FROM empleado LEFT JOIN departamento
ON empleado.id_departamento = departamento.id
```

2

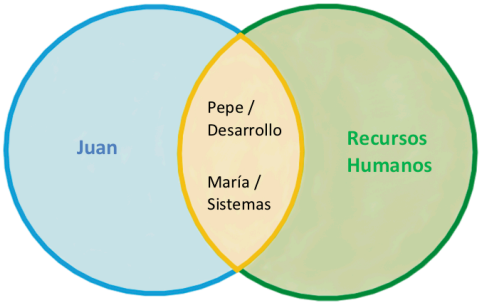
Tabla: empleado

id	nombre	id_departamento
1	Pepe	1
2	María	2
3	Juan	NULL

Tabla: departamento

id	nombre
1	Desarrollo
2	Sistemas
3	Recursos Humanos

Estas filas quedan **fuera de la intersección**



3

El **resultado de la operación LEFT JOIN** es:

empleado.id	empleado.nombre	empleado.id_departamento	departamento.id	departamento.nombre
1	Pepe	1	1	Desarrollo
2	María	2	2	Sistemas
3	Juan	NULL	NULL	NULL

Ejemplo de `RIGHT OUTER JOIN`:

```
SELECT *
FROM empleado RIGHT JOIN departamento
ON empleado.codigo_departamento = departamento.codigo
```

Esta consulta devolverá todas las filas de la tabla que hemos colocado a la derecha de la composición, en este caso la tabla `departamento`. Y relacionará las filas de la tabla de la derecha (`departamento`) con las filas de la tabla de la izquierda (`empleado`) con las que encuentre una coincidencia. Si no encuentra ninguna coincidencia, se mostrarán los valores de la fila de la tabla derecha (`departamento`) y en los valores de la tabla izquierda (`empleado`) donde no ha encontrado una coincidencia mostrará el valor `NULL`.

RIGHT JOIN

1

```
/* SQL 2 */
SELECT *
FROM empleado RIGHT JOIN departamento
ON empleado.id_departamento = departamento.id
```

2

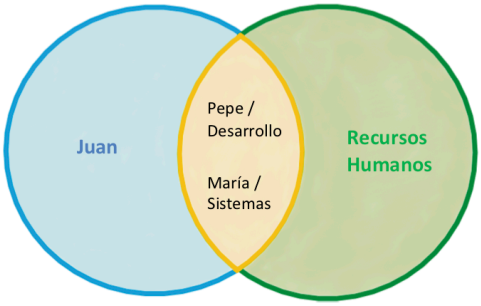
Tabla: empleado

id	nombre	id_departamento
1	Pepe	1
2	María	2
3	Juan	NULL

Tabla: departamento

id	nombre
1	Desarrollo
2	Sistemas
3	Recursos Humanos

Estas filas quedan **fuera de la intersección**



3

El **resultado de la operación RIGHT JOIN** es:

empleado.id	empleado.nombre	empleado.id_departamento	departamento.id	departamento.nombre
1	Pepe	1	1	Desarrollo
2	María	2	2	Sistemas
NULL	NULL	NULL	3	Recursos Humanos

Ejemplo de `FULL OUTER JOIN`:

La composición `FULL OUTER JOIN` **no está implementada en MySQL**, por lo tanto para poder simular esta operación será necesario hacer uso del operador `UNION`, que nos realiza la unión del resultado de dos consultas.

El resultado esperado de una composición de tipo `FULL OUTER JOIN` es obtener la intersección de las dos tablas, junto las filas de ambas tablas que no se puedan combinar. Dicho con otras palabras, el resultado sería el equivalente a realizar la union de una consulta de tipo `LEFT JOIN` y una consultas de tipo `RIGHT JOIN` sobre las mismas tablas.

```
SELECT *
FROM empleado LEFT JOIN departamento
ON empleado.codigo_departamento = departamento.codigo

UNION

SELECT *
FROM empleado RIGHT JOIN departamento
ON empleado.codigo_departamento = departamento.codigo
```

Para poder utilizar el operador `UNION` entre dos o más consultas deberá tener en cuenta que:

- Deben tener el mismo número de columnas.
- Las columnas que se van a unir tienen que tener tipos de datos similares.

Para ordenar los resultados tras aplicar una operación de `UNION` existen dos soluciones:

- Usar la posición de la columna sobre la que queremos ordenar los resultados en el `ORDER BY`.
- Crear un alias en las columnas del primer `SELECT` sobre la que queremos ordenar los resultados y usarlo en el `ORDER BY`.

Ejemplo:

```
/* Solución 1 */
SELECT departamento.nombre, empleado.apellido1, empleado.apellido2, empleado.
    nombre
FROM empleado LEFT JOIN departamento
ON empleado.codigo_departamento=departamento.codigo

UNION

SELECT departamento.nombre, empleado.apellido1, empleado.apellido2, empleado.
    nombre
FROM empleado RIGHT JOIN departamento
ON empleado.codigo_departamento=departamento.codigo
ORDER BY 1, 2, 3, 4;
```

```
/* Solución 2 */
SELECT
  departamento.nombre AS nombre_departamento,
  empleado.apellido1, empleado.apellido2, empleado.nombre
```

```
FROM empleado LEFT JOIN departamento
ON empleado.codigo_departamento=departamento.codigo

UNION

SELECT departamento.nombre, empleado.apellido1, empleado.apellido2, empleado.
    nombre
FROM empleado RIGHT JOIN departamento
ON empleado.codigo_departamento=departamento.codigo
ORDER BY nombre_departamento;
```

Ejemplo de `NATURAL LEFT JOIN`:

```
SELECT *
FROM empleado NATURAL LEFT JOIN departamento
```

Esta consulta realiza un `LEFT JOIN` entre las dos tablas, la única diferencia es que en este caso no es necesario utilizar la cláusula `ON` para indicar sobre qué columna vamos a relacionar las dos tablas. **En este caso las tablas se van a relacionar sobre aquellas columnas que tengan el mismo nombre**. Por lo tanto, sólo deberíamos utilizar una composición de tipo `NATURAL LEFT JOIN` cuando estemos seguros de que los nombres de las columnas sobre las que quiero relacionar las dos tablas se llaman igual en las dos tablas.

Capítulo 2

Errores comunes

1. Cuando estamos usando `LEFT JOIN` o `RIGHT JOIN` no deberíamos tener varias condiciones en la cláusula `ON`.

Consulta incorrecta

```
SELECT *
FROM fabricante LEFT JOIN producto
ON fabricante.codigo = producto.codigo_fabricante AND
   producto.codigo_fabricante IS NULL;
```

Consulta correcta.

```
SELECT *
FROM fabricante LEFT JOIN producto
ON fabricante.codigo = producto.codigo_fabricante
WHERE producto.codigo_fabricante IS NULL;
```

Capítulo 3

Referencias

- Wikibook SQL Exercises.
- Tutorial SQL de w3resource.
- MySQL Join Types by Steve Stedman.
- **Bases de Datos**. 2ª Edición. Grupo editorial Garceta. Iván López Montalbán, Manuel de Castro Vázquez y John Ospino Rivas.
- INNER JOIN.
- LEFT JOIN.
- RIGHT JOIN.

Capítulo 4

Licencia