Material descargable

MySQL: Estructura y Tipos de JOIN

egg

Introducción a los JOIN en MySQL

Los **JOIN en MySQL** son operaciones que se utilizan para combinar datos de dos o más tablas en una sola consulta.Hasta ahora, con las consultas multitablas veníamos haciendo JOINs implícitos. Cuando ejecutamos las querys estas eran tratadas como un JOIN permitiendo obtener la información de manera más eficiente gracias al trabajo interno del servidor de base de datos.

A partir de ahora es importante que cuando quieras hacer una consulta multitabla uses las sintaxis de la cláusula JOIN porque es mucho más legible y mantenible cuando se trabaja con otros programadores. ¡Vamos a por ello!

Estructura básica de un JOIN

La estructura básica de una sentencia SQL que utiliza un JOIN se ve de la siguiente manera:

```
SELECT *
FROM tabla1
JOIN tabla2
       ON tabla1.columna = tabla2.columna;
```

- **SELECT:** Especifica las columnas que deseas recuperar.
- **FROM:** Indica la tabla principal de la que se seleccionarán los datos.
- **JOIN:** Es la palabra clave que indica que se realizará una operación de JOIN.
- **tabla1 y tabla2:** Son las tablas que se unirán.

- **ON:** Es la condición de unión que especifica cómo se relacionan las tablas. Debes definir qué columnas de tabla1 y tabla2 deben coincidir para que se realice la unión.

Tipos de JOIN en MySQL

Existen varios tipos de JOIN en MySQL que te permiten controlar cómo se combinan las filas de las tablas. Los tipos de JOIN más comunes son:

- **INNER JOIN**

El **INNER JOIN** recupera filas de ambas tablas solo cuando hay una coincidencia entre las columnas especificadas en la condición de unión. Si no hay una coincidencia, las filas no se incluyen en el resultado.

```
SELECT clientes.nombre, pedidos.producto
FROM clientes
INNER JOIN pedidos
      ON clientes.id = pedidos.cliente_id;
```

💡*JOIN e INNER JOIN son equivalentes. Es una buena práctica escribir INNER JOIN en lugar de solo JOIN.*

- **LEFT JOIN (o LEFT OUTER JOIN)**

El **LEFT JOIN** recupera todas las filas de la tabla izquierda (primera tabla mencionada) y las filas coincidentes de la tabla derecha (segunda tabla mencionada). Si no hay una coincidencia en la tabla derecha, se devuelven valores NULL en las columnas de la tabla derecha.

```
SELECT e.nombre, d.nombre
FROM empleados e
LEFT JOIN departamentos d
      ON e.departamento_id = d.id;
```

Por ejemplo:

```
SELECT e.nombre, e.apellido, v.empleado_id
FROM empleados e
LEFT JOIN ventas v
          ON e.id = v.empleado_id group by e.id;
```

- **RIGHT JOIN (o RIGHT OUTER JOIN)**

El **RIGHT JOIN** es similar al LEFT JOIN, pero recupera todas las filas de la tabla derecha y las filas coincidentes de la tabla izquierda. Las filas de la tabla izquierda que no tienen coincidencias se devuelven con valores NULL.

```
SELECT e.nombre, d.nombre FROM empleados e RIGHT JOIN departamentos d ON
e.departamento_id = d.id;
```

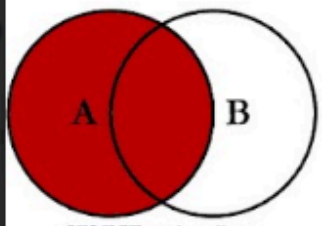
💡*El resultado de usar RIGHT JOIN es igual que el resultado de usar LEFT JOIN pero intercambiando de posición las tablas en la consulta. Se recomienda siempre usar LEFT JOIN por una cuestión de compatibilidad con otros servidores de bases de datos. La mayoría de los sistemas de gestión de bases de datos admiten LEFT JOIN pero algunos pueden no admitir RIGHT JOIN.*

Por ejemplo:

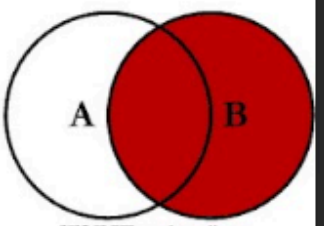
```
SELECT v.empleado_id, e.nombre, e.apellido  FROM ventas v RIGHT JOIN
empleados e ON e.id = v.empleado_id group by e.id;
```

A continuación, te presentamos una imagen que ilustra de manera gráfica los diversos tipos de joins en SQL utilizando JOINS. Esta representación visual te brindará una comprensión clara de cómo se establecen las relaciones entre tablas según la instrucción empleada:

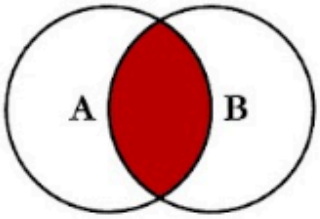
SQL JOINS



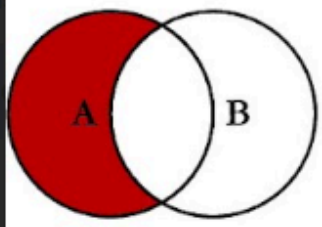

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```



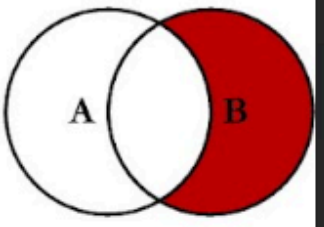

```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```



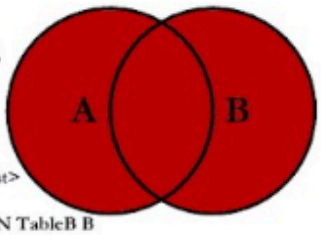

```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```



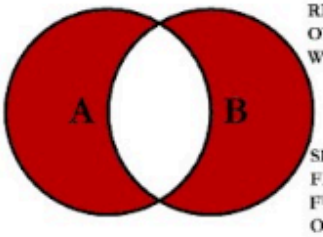

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```




```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
```




```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
```




```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
```