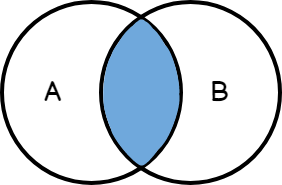
JOIN

Empezaremos con la definición. Join es el proceso de tomar datos de varias tablas y colocarlos en una vista generada. Por tanto, una instrucción de “SQL JOIN” en un comando Select combina las columnas entre una o más tablas en una base de datos relacional y retorna a un conjunto de datos.

“El FROM” también es parte esencial de la instrucción Select y es aquí donde se especifica de qué tabla estamos extrayendo los datos. La parte de join es donde queremos unir datos de varias tablas y tenemos tres tipos diferentes de combinaciones

* **Inner join** – esta es la opción predeterminada. Si no se especifica el tipo de unión, se establecerá de manera predeterminada como la unión interna. Esto implica que si estamos uniendo dos tablas en una columna común, solo retornaran los datos que coincidan en ambas tablas



Por ejemplo, si queremos listar a los empleados e indicar el nombre del departamento al que pertenecen, podemos hacer lo siguiente:

SELECT \*

FROM Empleados E

JOIN Departamentos D

ON E.DepartamentoId = D.Id

Con esto, nuestro resultado será:

| **Nombre** | **DepartmentoId** | **Id** | **Nombre** |
| --- | --- | --- | --- |
| Rafferty | 31 | 31 | Sales |
| Jones | 33 | 33 | Engineering |
| Heisenberg | 33 | 33 | Engineering |
| Robinson | 34 | 34 | Clerical |
| Smith | 34 | 34 | Clerical |

Y a partir de aquí podemos notar lo siguiente:

* El empleado "Williams" no aparece en los resultados, ya que no pertenece a ningún departamento existente.
* El departamento "Marketing" tampoco aparece, ya que ningún empleado pertenece a dicho departamento.

¿Por qué ocurre esto? Porque JOIN muestra como resultado la intersección de ambas tablas.

También hay que tener en cuenta que, en los resultados vemos 4 columnas. Las 2 primeras se corresponden con la tabla Empleados y las últimas con Departamentos.

Esto ocurre porque estamos seleccionando todas las columnas con un \*.

Si queremos, podemos ser específicos y seleccionar sólo 2 columnas (además de asignarles un alias):

SELECT

E.Nombre as 'Empleado',

D.Nombre as 'Departamento'

FROM Empleados E

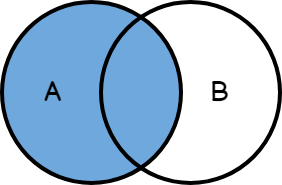
JOIN Departamentos D

ON E.DepartamentoId = D.Id

Y así obtener:

| **Empleado** | **Departamento** |
| --- | --- |
| Rafferty | Sales |
| Jones | Engineering |
| Heisenberg | Engineering |
| Robinson | Clerical |
| Smith | Clerical |

* **Left join** – este tipo de unión significa que solo retornan todos los datos de la tabla de la mano izquierda, solo si los datos coinciden con la tabla de la mano derecha



He aquí una consulta de ejemplo:

SELECT

E.Nombre as 'Empleado',

D.Nombre as 'Departamento'

FROM Empleados E

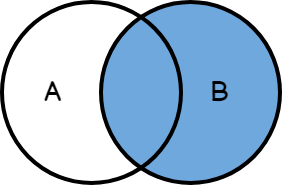
LEFT JOIN Departamentos D

ON E.DepartamentoId = D.Id

* La tabla Empleados es la primera tabla en aparecer en la consulta (en el FROM), por lo tanto ésta es la tabla LEFT (izquierda), y todas sus filas se mostrarán en los resultados.
* La tabla Departamentos es la tabla de la derecha (aparece luego del LEFT JOIN). Por lo tanto, si se encuentran coincidencias, se mostrarán los valores correspondientes, pero sino, aparecerá NULL en los resultados.

| **Empleado** | **Departamento** |
| --- | --- |
| Rafferty | Sales |
| Jones | Engineering |
| Heisenberg | Engineering |
| Robinson | Clerical |
| Smith | Clerical |
| Williams | NULL |

* **Right join**– este tipo de unión es el caso opuesto al anterior. Implica que solo retornaran los datos de la tabla de la mano derecha, solo si los datos coinciden con la tabla de la mano izquierda



De tal modo que si usamos la siguiente consulta, estaremos **mostrando todas las filas de la tabla de la derecha**:

SELECT

E.Nombre as 'Empleado',

D.Nombre as 'Departamento'

FROM Empleados E

RIGHT JOIN Departamentos D

ON E.DepartamentoId = D.Id

La tabla de la izquierda es Empleados, mientras que Departamentos es la tabla de la derecha.

La tabla asociada al FROM será siempre la tabla LEFT, y la tabla que viene después del JOIN será la tabla RIGHT.

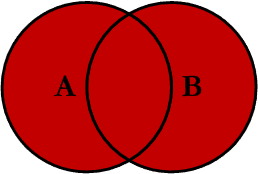
Entonces el resultado mostrará todos los departamentos al menos 1 vez.

Y si no hay ningún empleado trabajando en un departamento determinado, se mostrará NULL. Pero el departamento aparecerá de igual forma.

| **Empleado** | **Departamento** |
| --- | --- |
| Rafferty | Sales |
| Jones | Engineering |
| Heisenberg | Engineering |
| Robinson | Clerical |
| Smith | Clerical |
| NULL | Marketing |

**Cláusula FULL JOIN**

Mientras que LEFT JOIN muestra todas las filas de la tabla izquierda, y RIGHT JOIN muestra todas las correspondientes a la tabla derecha, FULL OUTER JOIN (o simplemente FULL JOIN) se encarga de mostrar todas las filas de ambas tablas, sin importar que no existan coincidencias (usará NULL como un valor por defecto para dichos casos).



Para nuestro ejemplo, ocurre lo siguiente:

SELECT

E.Nombre as 'Empleado',

D.Nombre as 'Departamento'

FROM Empleados E

FULL JOIN Departamentos D

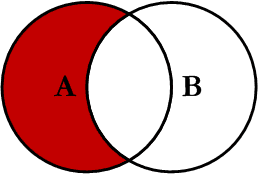
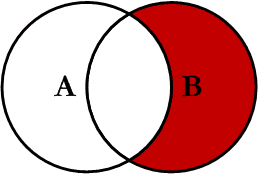
ON E.DepartamentoId = D.Id

Se muestra el empleado "Williams" a pesasr que no está asignado a ningún departamento, y se muestra el departamento de "Marketing" a pesar que aún nadie está trabajando allí:

| **Empleado** | **Departamento** |
| --- | --- |
| Rafferty | Sales |
| Jones | Engineering |
| Heisenberg | Engineering |
| Robinson | Clerical |
| Smith | Clerical |
| Williams | NULL |
| NULL | Marketing |

**Más variantes**

Si prestamos atención a los diagramas de Venn, vamos a notar que es posible formar incluso más combinaciones, al momento de seleccionar datos.

Por ejemplo, tenemos el siguiente caso, conocido como Left Excluding JOIN:Y de igual manera Right Excluding JOIN:

Estas combinaciones son posibles de lograr si añadimos algunas condiciones a nuestras consultas, haciendo uso de la cláusula WHERE.

Por ejemplo, siguiendo el ejemplo que estamos viendo (donde Empleados es la tabla izquierda y Departamentos la tabla derecha):

* Left Excluding JOIN nos permitirá obtener la lista de empleados que aún no han sido asignados a ningún departamento de trabajo.
* Mientras que Right Excluding JOIN nos mostrará la lista de departamentos que no tienen a ningún trabajador asociado