

Tema 6. DML – DQL. Consultas

Composición de tablas

Es muy habitual necesitar en una consulta datos que se encuentran distribuidos en varias tablas.

Las bases de datos relacionales se basan en que los datos se distribuyen en tablas que se pueden relacionar mediante un campo. Ese campo es el que permite integrar los datos de las tablas

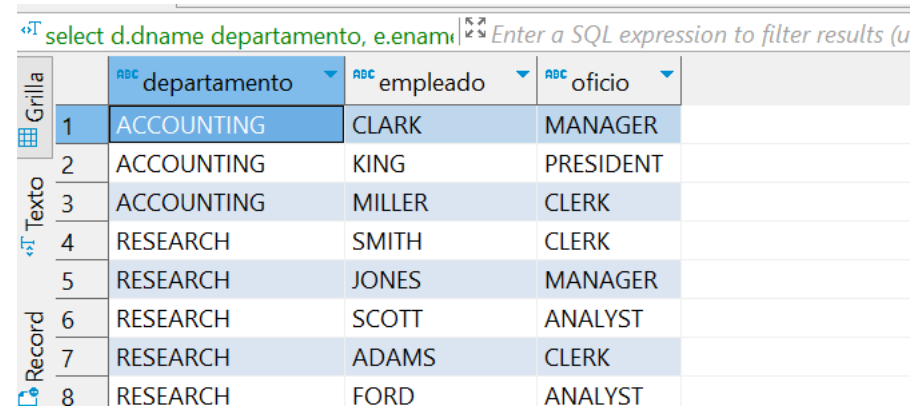
Se basa y se sostiene en todo lo estudiado hasta el momento de las referencias entre tablas mediante claves foráneas

Composición de tablas

Los nombres de las tablas aparecen varias veces en la sentencia SQL. Si, además, los nombres son largos, la escritura de las sentencias pasa a convertirse en algo largo y tedioso.

Para evitarlo, es recomendable utilizar ALIAS de tablas y de columnas:

```
select d.dname departamento, e.ename empleado, e.job oficio
from dept d, emp e
where d.deptno = e.DEPTNO
```



	departamento	empleado	oficio	
1	ACCOUNTING	CLARK	MANAGER	
2	ACCOUNTING	KING	PRESIDENT	
3	ACCOUNTING	MILLER	CLERK	
4	RESEARCH	SMITH	CLERK	
5	RESEARCH	JONES	MANAGER	
6	RESEARCH	SCOTT	ANALYST	
7	RESEARCH	ADAMS	CLERK	
8	RESEARCH	FORD	ANALYST	

Composición de tablas. NATURAL JOIN

Establece una relación de igualdad entre las tablas a través de los campos que tengan el mismo nombre en ambas tablas. Si hay dos o más columnas de las dos tablas que se llaman igual el JOIN se realiza teniendo en cuenta ambas columnas

Ejemplo: Mostrar un informe con el nombre, oficio, nombre de departamento y localidad de cada uno de los empleados del departamento:

```
SELECT e.ename Nombre, e.job Oficio, d.dname Departamento, D.loc  
Localidad  
FROM emp e NATURAL JOIN dept d;
```

Ojo. NATURAL JOIN -> NO es lo mismo que JOIN. Lo veremos más adelante.

Composición de tablas. NATURAL JOIN

Mostrar el nombre del departamento y el número de empleados que trabajan en él

```
select dname Departamento, count(*) empleados from emp natural join dept
group by dname
```

 `select dname Departamento, count(*)`  Enter a SQL expression to filter results (use Ctrl+Space)

Grilla	Departamento		empleados	
	1	ACCOUNTING	3	
	2	RESEARCH	5	
	3	SALES	6	
Texto				
Record				

Composición de tablas. JOIN USING

Si en una unión natural los campos de unión tienen el mismo nombre pero distintos tipos de dato provocan un error. Para evitar este problema se usa la cláusula USING. Esta cláusula permite definir el campo de unión entre las tablas.

Ejemplo. Mostrar el nombre de los empleados y la localización de su departamento para todos los empleados del departamento número 30:

```
SELECT ename, loc
FROM emp JOIN dept USING (deptno)
WHERE deptno=30;
```

Grilla		SELECT ename, loc FROM emp JOIN de	
cord		ABC ename	ABC loc
	1	ALLEN	CHICAGO
	2	WARD	CHICAGO
	3	MARTIN	CHICAGO
	4	BLAKE	CHICAGO
	5	TURNER	CHICAGO
	6	JAMES	CHICAGO

Composición de tablas. JOIN ON

No siempre la unión se realiza mediante un campo con el mismo nombre o con una condición de igualdad.

Para estos casos se tiene que usar la cláusula ON.

Establece relaciones cuya condición se establece manualmente, lo que permite realizar asociaciones más complejas o bien asociaciones cuyos campos en las tablas no tienen el mismo nombre

Composición de tablas. JOIN ON

Mostrar el nombre del empleado y el nombre del departamento al que pertenece para todos los empleados que ganan más de 2500

```
select ename nombre, dname departamento
from emp e join dept d on e.deptno = d.deptno
where e.sal > 2500
```

select ename nombre, dname departar Enter a SQL expression to filter results

	nombre	departamento	
1	JONES	RESEARCH	
2	BLAKE	SALES	
3	SCOTT	RESEARCH	
4	KING	ACCOUNTING	
5	FORD	RESEARCH	

Composición de tablas. JOIN ON

Mostrar el nombre, salario y la categoría salarial para los empleados del departamento 20.

****Introducimos una tabla más en nuestro esquema de tablas de emp y dept, ahora también va a estar la tabla: SALGRADE

```
select ename nombre, sal salario, grade  
from emp e join salgrade s on (e.sal between s.losal and s.hisal)  
where deptno = 20
```

select ename nombre, sal salario, grade | Enter a SQL expression to filter results

	nombre	salario	grade
1	ADAMS	1.100	1
2	SMITH	800	1
3	FORD	3.000	4
4	SCOTT	3.000	4
5	JONES	2.975	4

Concatenación de tablas. Cross JOIN

Realiza un producto cruzado entre las tablas indicadas. Eso significa que cada tupla de la primera tabla se combina con cada tupla de la segunda tabla. No es una operación muy utilizada, aunque posibilita resolver consultas de alto nivel de complejidad.

Mostrar el producto cartesiano de la tabla empleados y departamentos:

```
select * from emp cross join dept:
```

En total 14 empleados x 4 departamentos = 56 filas

	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO	deptno	DNAME
1	7.369	SMITH	CLERK	7.902	1980-12-17	800	[NULL]	20	40	OPERATIONS
2	7.369	SMITH	CLERK	7.902	1980-12-17	800	[NULL]	20	30	SALES
3	7.369	SMITH	CLERK	7.902	1980-12-17	800	[NULL]	20	20	RESEARCH
4	7.369	SMITH	CLERK	7.902	1980-12-17	800	[NULL]	20	10	ACCOUNTING
5	7.499	ALLEN	SALESMAN	7.698	1981-02-20	1.600	300	30	40	OPERATIONS
6	7.499	ALLEN	SALESMAN	7.698	1981-02-20	1.600	300	30	30	SALES
7	7.499	ALLEN	SALESMAN	7.698	1981-02-20	1.600	300	30	20	RESEARCH

Renovar Save Cancel Exportar datos ... 200 56

56 row(s) fetched - 4ms (3ms fetch), on 2024-03-31 at 19:28:12

Concatenación de tablas. Cross JOIN

Conseguir un listado de las parejas de empleados (MANAGER, CLERK) que la suma de sus salarios sea inferior a 4000

```
SELECT E1.ename Empleado1, E1.job , E1.sal SalarioE1,  
E2.ename Empleado2, E2.job OficioE2, E2.sal SalarioE2,  
E1.sal + E2.sal TOTAL  
FROM emp E1 CROSS JOIN emp E2  
WHERE E1.sal+E2.sal < 4000 AND E1.job='MANAGER' AND E2.job='CLERK';
```

SQL Editor: `SELECT E1.ename Empleado1, E1.job ,` | *Enter a SQL expression to filter results (use Ctrl+Space)*

	Empleado1	job	SalarioE1	Empleado2	OficioE2	SalarioE2	TOTAL
1	CLARK	MANAGER	2.450	SMITH	CLERK	800	3.250
2	BLAKE	MANAGER	2.850	SMITH	CLERK	800	3.650
3	JONES	MANAGER	2.975	SMITH	CLERK	800	3.775
4	CLARK	MANAGER	2.450	ADAMS	CLERK	1.100	3.550
5	BLAKE	MANAGER	2.850	ADAMS	CLERK	1.100	3.950
6	CLARK	MANAGER	2.450	JAMES	CLERK	950	3.400
7	BLAKE	MANAGER	2.850	JAMES	CLERK	950	3.800
8	JONES	MANAGER	2.975	JAMES	CLERK	950	3.925

Concatenación de tablas. Cross JOIN

Listado de las parejas de empleados (MANAGER, CLERK) que trabajan en NEW YORK

```
SELECT E1.ename 'EMPLEADO 1', E1.job 'JOB E1',  
E2.ename 'EMPLEADO 2', E2.job 'JOB E2',  
D1.loc 'CIUDAD'  
FROM (emp E1 natural Join dept D1) CROSS JOIN (emp E2 natural JOIN dept  
D2)  
WHERE E1.job = 'MANAGER' and E2.job='CLERK' and D1.loc = 'NEW YORK' and  
d2.loc = 'NEW YORK';
```

SELECT E1.ename 'EMPLEADO 1', E1.job 'JOB E1', E2.ename 'EMPLEADO 2', E2.job 'JOB E2', D1.loc 'CIUDAD'						
Enter a SQL expression to filter results (use Ctrl+Space)						
	EMPLEADO 1	JOB E1	EMPLEADO 2	JOB E2	CIUDAD	
1	CLARK	MANAGER	MILLER	CLERK	NEW YORK	

Concatenación de tablas.

Concatenar dos tablas entre sí, luego, concatena el resultado con la tercera tabla

Mostrar el nombre de empleado, nombre del departamento al que pertenece y la categoría salarial de todos los empleados cuyo oficio sea 'CLERK'

```
select e1.ename nombre, d.dname departamento, s.grade categoria
from (emp e1 natural join dept d)
cross join salgrade s on (e1.sal between s.losal and s.hisal)
where e1.job = 'CLERK' ##Funciona igual con cross que sin cross
```

Concatenación externa de tablas.

En ocasiones resulta muy interesante que no se pierda ninguna fila de una u otra tabla al realizar la concatenación. Para evitarlo es necesario hacer una concatenación externa. Esta es muy similar a la interna sólo que evita que se pierdan filas que no están relacionadas.

Por ejemplo, se desea mostrar un informe en el que aparezca el nombre del departamento y el número de empleados que trabajan en ese departamento:

```
SELECT D.dname, COUNT(*) as 'NumEmp'  
FROM emp E JOIN dept D ON (E.deptno = D.deptno)  
GROUP BY D.dname;
```

	ABC dname ▼	123 NumEmp ▼
1	ACCOUNTING	3
2	RESEARCH	5
3	SALES	6

Continuación en la siguiente diapositiva

Concatenación externa de tablas.

Se observa como en el listado no aparece el departamento 40 (OPERATIONS) que no hay ningún empleado en la tabla EMP cuyo deptno sea el 40. Sin embargo, el informe no estará completo hasta que no aparezca una fila con dicho departamento:

	ABC dname ▼	123 NumEmp ▼	
1	ACCOUNTING	3	
2	RESEARCH	5	
3	SALES	6	
4	OPERATIONS	0	

¿CÓMO SE HA LLEGADO HASTA ESTE RESULTADO?

Concatenación externa de tablas.

LEFT OUTER JOIN

El resultado contiene todas las filas de la TABLA 1.

Las filas de la TABLA 1 que se relacionan con alguna de las filas de la TABLA 2 aparecen concatenadas en el resultado

Las filas de la TABLA 1 que no se relacionan con ninguna fila de la TABLA 2 aparecen en el resultado concatenadas con una fila de nulos.

Ejemplo. Siguiendo con el caso anterior, para listar el nombre del departamento y el número de personas que trabajan en él incluyendo el departamento 'OPERATIONS' tendríamos que contar el número de registros que tiene el campo ename teniendo en cuenta todos los departamentos:

```
SELECT D.dname, COUNT(E.ename) "NUMEMP"  
FROM dept D LEFT OUTER JOIN emp E ON (D.deptno = E.deptno)  
GROUP BY D.dname;
```

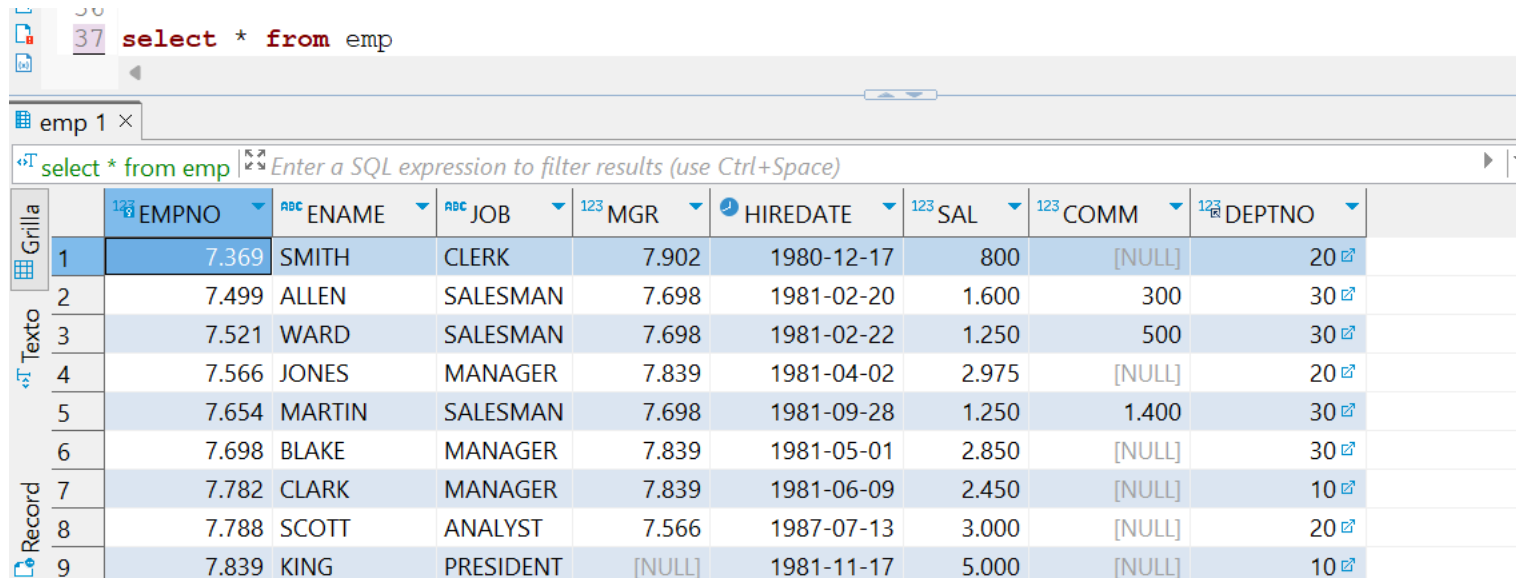
	ABC dname ▼	123 NumEmp ▼
1	ACCOUNTING	3
2	RESEARCH	5
3	SALES	6
4	OPERATIONS	0

Concatenación externa de tablas.

LEFT OUTER JOIN

Ejemplo. Mostrar un listado con el nombre del empleado y el nombre del superior. Si no tienen superior, como por ejemplo, el presidente de la compañía el campo superior debe aparecer a NULL.

Para realizar la consulta primero hay que ver en cual es el contenido de la tabla EMP



The screenshot shows a database application window. At the top, a SQL query is entered: `select * from emp`. Below the query, a tab labeled 'emp 1' is active. The results are displayed in a table with columns: EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, COMM, and DEPTNO. The table contains 9 rows of data, including the president (KING) who has a NULL value for the MGR field.

	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	7.369	SMITH	CLERK	7.902	1980-12-17	800	[NULL]	20
2	7.499	ALLEN	SALESMAN	7.698	1981-02-20	1.600	300	30
3	7.521	WARD	SALESMAN	7.698	1981-02-22	1.250	500	30
4	7.566	JONES	MANAGER	7.839	1981-04-02	2.975	[NULL]	20
5	7.654	MARTIN	SALESMAN	7.698	1981-09-28	1.250	1.400	30
6	7.698	BLAKE	MANAGER	7.839	1981-05-01	2.850	[NULL]	30
7	7.782	CLARK	MANAGER	7.839	1981-06-09	2.450	[NULL]	10
8	7.788	SCOTT	ANALYST	7.566	1987-07-13	3.000	[NULL]	20
9	7.839	KING	PRESIDENT	[NULL]	1981-11-17	5.000	[NULL]	10

Concatenación externa de tablas.

LEFT OUTER JOIN

Ejemplo. Continuación. Como se puede ver, el empleado 'KING' es el presidente de la compañía con lo cual no tiene superior (campo MGR a NULL). En una consulta JOIN no saldría 'KING' ya que al hacer la unión no tendría correspondencia

```
SELECT E.ename "EMPLEADO", S.ename "SUPERIOR"  
FROM emp E JOIN emp S ON (E.mgr = S.empno) ;
```

EMPLEADO	SUPERIOR	
SMITH	FORD	
ALLEN	BLAKE	
WARD	BLAKE	
JONES	KING	
MARTIN	BLAKE	
BLAKE	KING	
CLARK	KING	
SCOTT	JONES	
TURNER	BLAKE	

Concatenación externa de tablas.

LEFT OUTER JOIN

Ejemplo. Continuación.

```
SELECT E.ename "EMPLEADO", S.ename "SUPERIOR"  
FROM emp E LEFT OUTER JOIN emp S ON (E.mgr = S.empno) ;
```

ABC EMPLEADO ▼	ABC SUPERIOR ▼	
SMITH	FORD	
ALLEN	BLAKE	
WARD	BLAKE	
JONES	KING	
MARTIN	BLAKE	
BLAKE	KING	
CLARK	KING	
SCOTT	JONES	
KING	[NULL]	

Concatenación externa de tablas.

LEFT OUTER JOIN

Ejemplo. Continuación.

```
SELECT E.ename "EMPLEADO", S.ename "SUPERIOR"  
FROM emp E LEFT OUTER JOIN emp S ON (E.mgr = S.empno) ;
```

ABC EMPLEADO ▼	ABC SUPERIOR ▼	
SMITH	FORD	
ALLEN	BLAKE	
WARD	BLAKE	
JONES	KING	
MARTIN	BLAKE	
BLAKE	KING	
CLARK	KING	
SCOTT	JONES	
KING	[NULL]	

Concatenación externa de tablas.

RIGHT OUTER JOIN

El resultado contiene todas las filas de la TABLA 2.

Las filas de la TABLA 2 que se relacionan con alguna de las filas de la TABLA 1 aparecen concatenadas en el resultado

Las filas de la TABLA 2 que no se relacionan con ninguna fila de la TABLA 1 aparecen en el resultado concatenadas con una fila de nulos.

Análogo a LEFT OUTER JOIN. Ejemplo del caso del número de empleados por departamento, se invierte el orden de las tablas y en vez de LEFT, se hace con RIGHT

```
SELECT D.dname, COUNT(E.ename) "NUMEMP"  
FROM emp e RIGHT OUTER JOIN dept d ON (D.deptno = E.deptno)  
GROUP BY D.dname;
```

A LEFT OUTER JOIN B es equivalente a B RIGHT OUTER JOIN A.

A RIGHT OUTER JOIN B es equivalente a B LEFT OUTER JOIN A.

Combinaciones especiales. UNION

La unión de dos tablas es otra tabla que contiene los registros que están en la TABLA 1, en la TABLA 2, o en ambas, eliminándose los registros duplicados.

Las tablas deben ser unión-compatibles, es decir, definidas sobre el mismo conjunto de atributos (mismas columnas) para obtener resultados con sentido. La consulta funcionará con SELECT realizadas sobre el mismo número de columnas

*UNION ALL. Igual que UNION pero muestra datos duplicados.

Ingenieros			Jefes			Ingenieros \cup Jefes		
E#	Nombre	Edad	E#	Nombre	Edad	E#	Nombre	Edad
320	José	34	320	José	34	320	José	34
322	Rosa	37	421	Jorge	48	322	Rosa	37
323	María	25				•	María	25
						421	Jorge	48

Combinaciones especiales. UNION

TABLA A	TABLA B
10	0
10	10
20	10
20	20
50	

A UNION B
0
10
20
50

A UN.ALL B
10
10
20
20
50
0
10
10
20

Combinaciones especiales. INTERSECT

Permite unir dos consultas SELECT de modo que el resultado serán las filas que estén presentes en ambas consultas.

*INTERSECT ALL. Si una misma fila aparece m veces en la primera sentencia y n veces en la segunda, en el resultado aparecerá esta fila $\min(m,n)$ veces

TABLA A	TABLA B
10	0
10	10
20	10
20	20
50	

A INTER B
10
20

A INT.ALL B
10
10
20

Combinaciones especiales. EXCEPT

Muestra los registros de la primera SELECT que no estén presentes en la segunda

TABLA A	TABLA B
10	0
10	10
20	10
20	20
50	

A MINUS B
50

Combinaciones especiales. Ejemplos

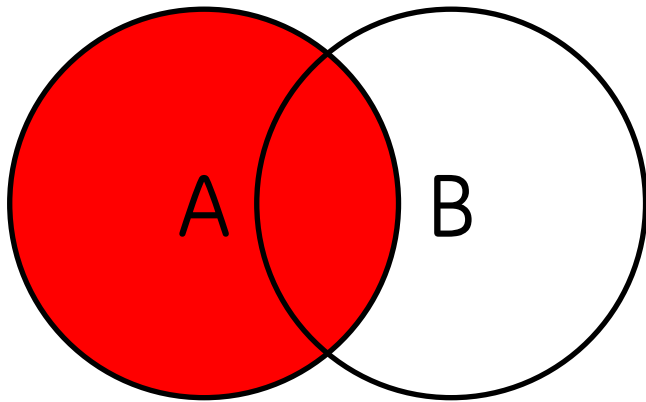
```
select DEPTNO from dept union select DEPTNO from EMP
```

```
select DEPTNO from dept intersect select DEPTNO from EMP
```

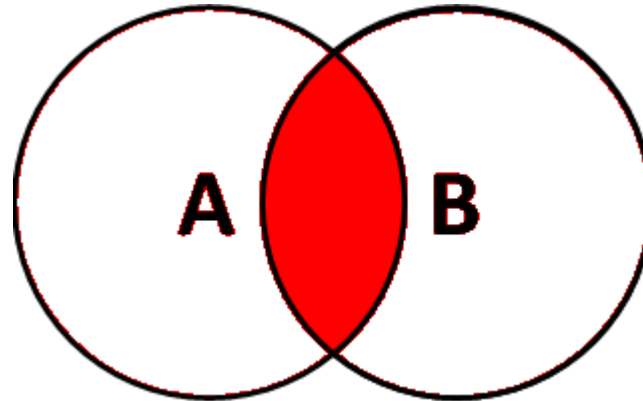
```
select DEPTNO from dept except select DEPTNO from EMP
```

*Cuando se realiza una operación algebraica de dos consultas, es obligatorio que coincida el número y tipo de las columnas devueltas por cada una de las consultas.

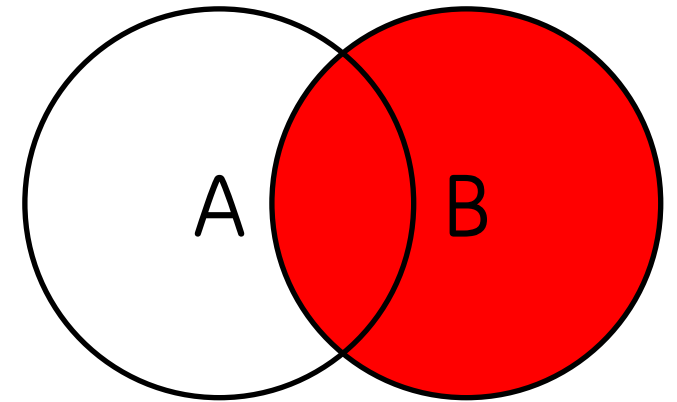
Resumen



SELECT <list> FROM A
LEFT OUTER JOIN B
ON A.key=B.key

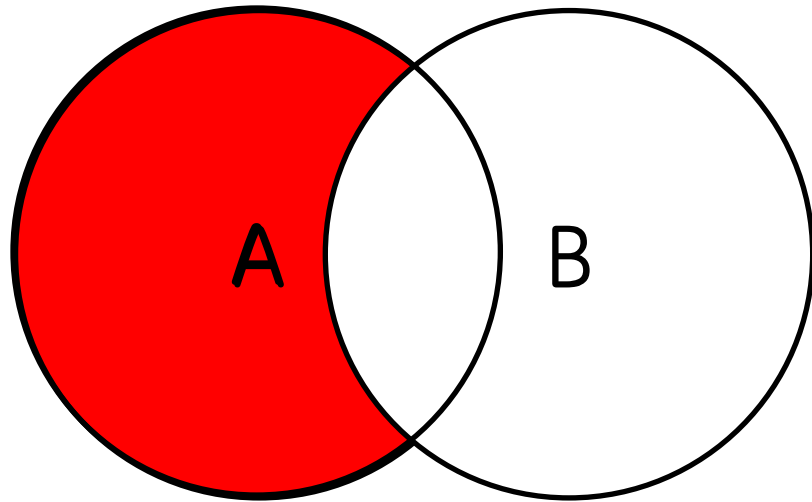


SELECT <list> FROM A
JOIN B
ON A.key=B.key

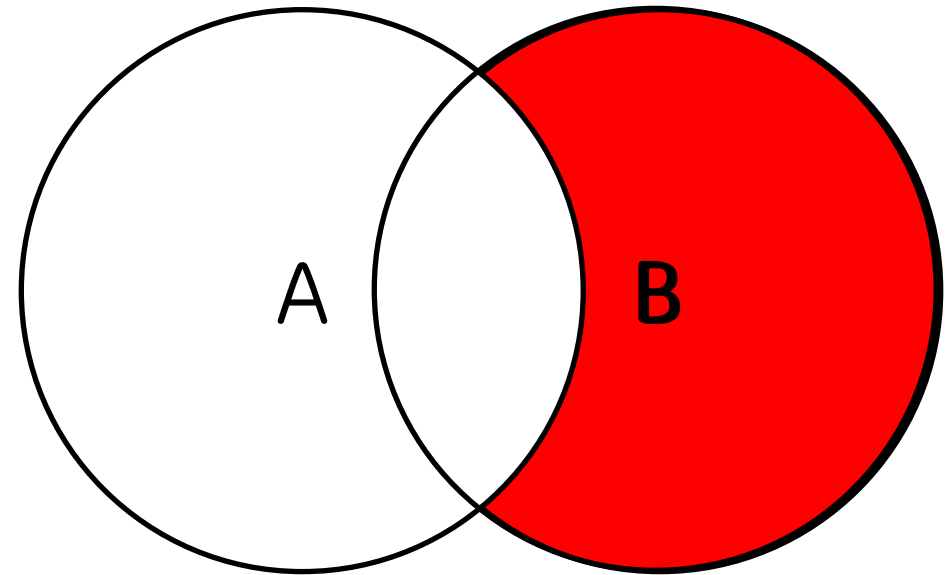


SELECT <list> FROM A
RIGHT OUTER JOIN B
ON A.key=B.key

Resumen

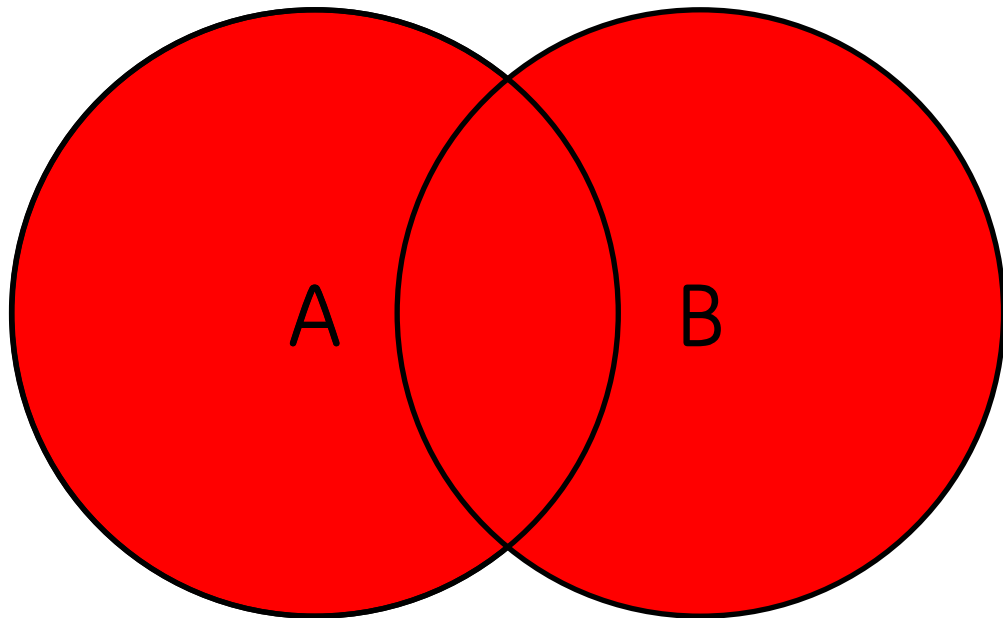


```
SELECT <list> FROM A  
LEFT OUTER JOIN B  
ON A.key=B.key  
WHERE B.key IS NULL
```

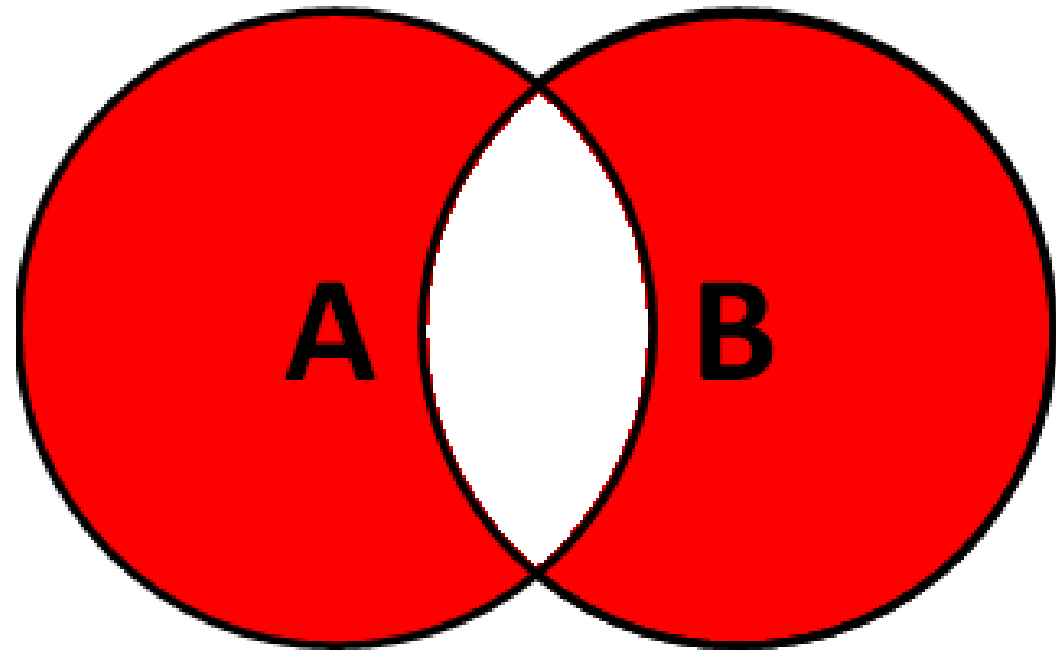


```
SELECT <list> FROM A  
RIGHT JOIN B  
ON A.key=B.key  
WHERE A.key IS NULL
```

Resumen



```
SELECT <list> FROM A  
FULL OUTER JOIN B  
ON A.key=B.key
```



```
SELECT <list> FROM A  
FULL OUTER JOIN B  
ON A.key=B.key  
WHERE A.key IS NULL  
OR B.Key IS NULL
```