Sentencia SELECT para consulta de datos

```
SELECT [DISTINCT] <lista_campos>
        FROM <lista_tablas>
        [WHERE <condición>]
        [GROUP BY <lista_campos> [HAVING <condición_agrupación>]]
        [ORDER BY <lista_campos>];
```


<lista_tablas> puede incluir una o varias tablas. En primer lugar, veremos consultas sobre una única tabla.

Para los ejemplos usaremos la siguiente base de datos:

```
MariaDB [company]> SHOW TABLES;
+------------------+
| Tables_in_company |
+------------------+
| dept              |
| emp               |
+------------------+
2 rows in set (0,001 sec)

MariaDB [company]> DESCRIBE dept;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| deptno | int(2)      | NO   | PRI | NULL    |       |
| dname  | varchar(14) | YES  |     | NULL    |       |
| loc    | varchar(13) | YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
3 rows in set (0,001 sec)

MariaDB [company]> DESCRIBE emp;
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| empno    | int(4)       | NO   | PRI | NULL    |       |
| ename    | varchar(10)  | YES  |     | NULL    |       |
| job      | varchar(9)   | YES  |     | NULL    |       |
| mgr      | int(4)       | YES  | MUL | NULL    |       |
| hiredate | date         | YES  |     | NULL    |       |
| sal      | decimal(7,2) | YES  |     | NULL    |       |
| comm     | decimal(7,2) | YES  |     | NULL    |       |
| deptno   | int(2)       | YES  | MUL | NULL    |       |
+----------+--------------+------+-----+---------+-------+
8 rows in set (0,001 sec)
```

1. Consultas sobre una tabla

<lista_campos> especifica la lista de campos o expresiones a obtener, separados por comas y en el orden especificado. Podemos usar * para referirnos a todos los campos de una tabla. DISTINCT elimina los valores repetidos.

<condición> puede ser una condición simple, o la unión de varias mediante operadores lógicos. Las condiciones generalmente comparan el valor de un campo con el de una variable, otro campo o una expresión.

Los operadores disponibles son:

- Comparativos: <, <=, >, >=, <>, =.
  - `SELECT * FROM dept WHERE deptno <> 10;`

```
+--------+------------+---------+
| deptno | dname      | loc     |
+--------+------------+---------+
|     20 | RESEARCH   | DALLAS  |
|     30 | SALES      | CHICAGO |
|     40 | OPERATIONS | BOSTON  |
+--------+------------+---------+
```

- BETWEEN … AND… Establece una comparación dentro de un intervalo (incluye los extremos del intervalo).
  - `SELECT * FROM dept WHERE deptno BETWEEN 20 AND 30;`

```
+--------+----------+---------+
| deptno | dname    | loc     |
+--------+----------+---------+
|     20 | RESEARCH | DALLAS  |
|     30 | SALES    | CHICAGO |
+--------+----------+---------+
```

- Lógicos: AND, OR, NOT.
  - `SELECT dname FROM dept WHERE deptno = 10 OR loc = 'DALLAS';`

```
+------------+
| dname      |
+------------+
| ACCOUNTING |
| RESEARCH   |
+------------+
```

- La función CONCAT concatena cadenas de caracteres provenientes de campos especificados en la consulta. Lo usamos para mostrar información de varias fuentes en una sola expresión.
  - `SELECT CONCAT(dname,' - ',loc) FROM dept;`

```
+-------------------------+
| CONCAT(dname,' - ',loc) |
+-------------------------+
| ACCOUNTING - NEW YORK   |
| RESEARCH - DALLAS       |
| SALES - CHICAGO         |
| OPERATIONS - BOSTON     |
+-------------------------+
```


- El operador IS NULL comprueba si un valor de un campo es nulo.
  - `SELECT * FROM emp WHERE comm IS NULL;`

```
+-------+--------+-----------+------+------------+---------+------+--------+
| empno | ename  | job       | mgr  | hiredate   | sal     | comm | deptno |
+-------+--------+-----------+------+------------+---------+------+--------+
|  7369 | SMITH  | CLERK     | 7902 | 1980-12-17 |  800.00 | NULL |     20 |
|  7566 | JONES  | MANAGER   | 7839 | 1981-04-02 | 2975.00 | NULL |     20 |
|  7698 | BLAKE  | MANAGER   | 7839 | 1981-05-01 | 2850.00 | NULL |     30 |
|  7782 | CLARK  | MANAGER   | 7839 | 1981-06-09 | 2450.00 | NULL |     10 |
|  7788 | SCOTT  | ANALYST   | 7566 | 1987-07-13 | 3000.00 | NULL |     20 |
|  7839 | KING   | PRESIDENT | NULL | 1981-11-17 | 5000.00 | NULL |     10 |
|  7876 | ADAMS  | CLERK     | 7788 | 1987-07-13 | 1100.00 | NULL |     20 |
|  7900 | JAMES  | CLERK     | 7698 | 1981-12-03 |  950.00 | NULL |     30 |
|  7902 | FORD   | ANALYST   | 7566 | 1981-12-03 | 3000.00 | NULL |     20 |
|  7934 | MILLER | CLERK     | 7782 | 1982-01-23 | 1300.00 | NULL |     10 |
+-------+--------+-----------+------+------------+---------+------+--------+
```

  - `SELECT * FROM emp WHERE comm IS NOT NULL;`

```
+-------+--------+----------+------+------------+---------+---------+--------+
| empno | ename  | job      | mgr  | hiredate   | sal     | comm    | deptno |
+-------+--------+----------+------+------------+---------+---------+--------+
|  7499 | ALLEN  | SALESMAN | 7698 | 1981-02-20 | 1600.00 |  300.00 |     30 |
|  7521 | WARD   | SALESMAN | 7698 | 1981-02-22 | 1250.00 |  500.00 |     30 |
|  7654 | MARTIN | SALESMAN | 7698 | 1981-09-28 | 1250.00 | 1400.00 |     30 |
|  7844 | TURNER | SALESMAN | 7698 | 1981-09-08 | 1500.00 |    0.00 |     30 |
+-------+--------+----------+------+------------+---------+---------+--------+
```

- El operador LIKE permite establecer un patrón para realizar comparaciones. Se usan los comodines % (cualquier cadena de caracteres) y _ (un único carácter). En ciertos SGBD se sustituye el % por * y el _ por ?
  - `SELECT * FROM emp WHERE ename LIKE '%ER';`

empno	ename	job	mgr	hiredate	sal	comm	deptno
7844	TURNER	SALESMAN	7698	1981-09-08	1500.00	0.00	30
7934	MILLER	CLERK	7782	1982-01-23	1300.00	NULL	10

2. Funciones de agrupamiento

Sobre <lista_campos> podemos realizar funciones que nos permiten hacer operaciones sobre los resultados obtenidos:

- Con la función COUNT se devuelve el número de registros que cumplen con la condición. Se puede especificar * para indicar todos los registros a contar, o bien podemos indicar un campo en concreto, del que se descontarán los valores NULL.
  - `SELECT COUNT(*) FROM emp;`

COUNT(*)
14

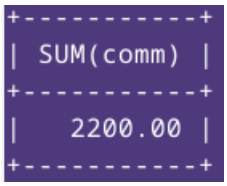
  - `SELECT COUNT(mgr) FROM emp;`

COUNT(mgr)
13

- La función AVG calcula la media aritmética de un campo numérico.
  - `SELECT AVG(sal) FROM emp;`

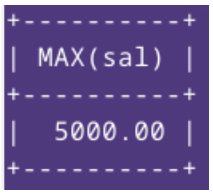
AVG(sal)
2073.214286

- La función SUM calcula el total de sumar los campos numéricos devueltos por la consulta.
  - `SELECT SUM(comm) FROM emp;`



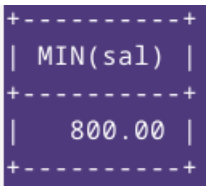
```
+-----------+
| SUM(comm) |
+-----------+
|   2200.00 |
+-----------+
```


- La función MAX retorna el valor máximo de entre los resultados de la consulta.
  - `SELECT MAX(sal) FROM emp;`



```
+----------+
| MAX(sal) |
+----------+
|  5000.00 |
+----------+
```

- La función MIN retorna el valor mínimo de entre los resultados de la consulta.
  - `SELECT MIN(sal) FROM emp;`



```
+----------+
| MIN(sal) |
+----------+
|   800.00 |
+----------+
```

- La cláusula GROUP BY nos permite agrupar los resultados por los valores obtenidos de la consulta:
  - `SELECT job FROM emp GROUP BY job;`

```
+-----------+
| job       |
+-----------+
| ANALYST   |
| CLERK     |
| MANAGER   |
| PRESIDENT |
| SALESMAN  |
+-----------+
```

- `SELECT COUNT(job) FROM emp GROUP BY job;`

```
+------------+
| COUNT(job) |
+------------+
|          2 |
|          4 |
|          3 |
|          1 |
|          4 |
+------------+
```

- `SELECT job, COUNT(job) FROM emp GROUP BY job;`

```
+-----------+------------+
| job       | COUNT(job) |
+-----------+------------+
| ANALYST   |          2 |
| CLERK     |          4 |
| MANAGER   |          3 |
| PRESIDENT |          1 |
| SALESMAN  |          4 |
+-----------+------------+
```

- La cláusula HAVING actúa sobre GROUP BY del mismo modo que WHERE sobre SELECT: impone a los resultados obtenidos una condición sobre la que se filtran éstos.
  - `SELECT job, COUNT(job), AVG(sal) FROM emp GROUP BY job HAVING AVG(sal)>2000;`

```
+-----------+------------+-------------+
| job       | COUNT(job) | AVG(sal)    |
+-----------+------------+-------------+
| ANALYST   |          2 | 3000.000000 |
| MANAGER   |          3 | 2758.333333 |
| PRESIDENT |          1 | 5000.000000 |
+-----------+------------+-------------+
```

- La cláusula ORDER BY permite especificar el orden en el que ordenamos los resultados obtenidos. Si indicamos varios campos, ordenará en primer lugar por el primero indicado, y dentro de éste por el segundo y así sucesivamente. Podemos especificar que el ordenamiento sea ascendente o descendente.
  - `SELECT * FROM emp ORDER BY deptno, sal DESC;`

empno	ename	job	mgr	hiredate	sal	comm	deptno
7839	KING	PRESIDENT	NULL	1981-11-17	5000.00	NULL	10
7782	CLARK	MANAGER	7839	1981-06-09	2450.00	NULL	10
7934	MILLER	CLERK	7782	1982-01-23	1300.00	NULL	10
7902	FORD	ANALYST	7566	1981-12-03	3000.00	NULL	20
7788	SCOTT	ANALYST	7566	1987-07-13	3000.00	NULL	20
7566	JONES	MANAGER	7839	1981-04-02	2975.00	NULL	20
7876	ADAMS	CLERK	7788	1987-07-13	1100.00	NULL	20
7369	SMITH	CLERK	7902	1980-12-17	800.00	NULL	20
7698	BLAKE	MANAGER	7839	1981-05-01	2850.00	NULL	30
7499	ALLEN	SALESMAN	7698	1981-02-20	1600.00	300.00	30
7844	TURNER	SALESMAN	7698	1981-09-08	1500.00	0.00	30
7654	MARTIN	SALESMAN	7698	1981-09-28	1250.00	1400.00	30
7521	WARD	SALESMAN	7698	1981-02-22	1250.00	500.00	30
7900	JAMES	CLERK	7698	1981-12-03	950.00	NULL	30

3. Consultas sobre varias tablas

- Podemos elaborar resultados provenientes de varias tablas relacionándolas mediante la cláusula WHERE:
  - `SELECT ename,job, loc FROM emp, dept WHERE emp.deptno = dept.deptno;`

ename	job	loc
CLARK	MANAGER	NEW YORK
KING	PRESIDENT	NEW YORK
MILLER	CLERK	NEW YORK
SMITH	CLERK	DALLAS
JONES	MANAGER	DALLAS
SCOTT	ANALYST	DALLAS
ADAMS	CLERK	DALLAS
FORD	ANALYST	DALLAS
ALLEN	SALESMAN	CHICAGO
WARD	SALESMAN	CHICAGO
MARTIN	SALESMAN	CHICAGO
BLAKE	MANAGER	CHICAGO
TURNER	SALESMAN	CHICAGO
JAMES	CLERK	CHICAGO

En caso de que existan nombres de campos iguales en varias tablas, tendremos que indicar a cual estamos referenciando indicándolo mediante: nombre_tabla.nombre_campo.

También podemos combinar el resultado de varias consultas SQL independientes mediante los siguientes operadores:

- El operador UNION [ALL] combina todos los datos que devuelve una consulta con todos los que devuelve otra. Por defecto, excluye los resultados repetidos, comportamiento que podemos evitar usando ALL.
  - `SELECT ename, job, loc FROM emp, dept WHERE emp.deptno = dept.deptno AND dept.deptno=10 UNION SELECT ename, job, loc FROM emp, dept WHERE emp.deptno = dept.deptno AND dept.deptno=20;`

ename	job	loc
CLARK	MANAGER	NEW YORK
KING	PRESIDENT	NEW YORK
MILLER	CLERK	NEW YORK
SMITH	CLERK	DALLAS
JONES	MANAGER	DALLAS
SCOTT	ANALYST	DALLAS
ADAMS	CLERK	DALLAS
FORD	ANALYST	DALLAS

- El operador EXCEPT retorna todos los resultados que están únicamente en la primera consulta, quitando todos los valores de la segunda. En caso de que se trate de conjuntos disjuntos, devolverá solo los resultados de la primera consulta.
  - `SELECT loc FROM dept WHERE deptno!=10 EXCEPT SELECT loc FROM dept WHERE deptno!=20;`

loc
DALLAS

- El operador INTERSECT retorna todos los resultados que son compartidos por ambas consultas, quitando todos los valores que están en una pero no en la otra.
  - `SELECT ename, job, loc FROM emp, dept WHERE emp.deptno = dept.deptno AND dept.deptno>=20 INTERSECT`
  - `SELECT ename, job, loc FROM emp, dept WHERE emp.deptno = dept.deptno AND dept.deptno<=20;`

ename	job	loc
SMITH	CLERK	DALLAS
JONES	MANAGER	DALLAS
SCOTT	ANALYST	DALLAS
ADAMS	CLERK	DALLAS
FORD	ANALYST	DALLAS

4. Subconsultas

Con SQL también podemos combinar consultas mediante la evaluación de los resultados de subconsultas.

A su vez, las subconsultas también pueden tener subconsultas anidadas. Pero no es recomendable abusar del anidamiento de subconsultas, pues tiene un coste elevado sobre el rendimiento.

Los operadores usados en subconsultas son:

- Operadores de comparación (>, =, <, etc). Se usan cuando estamos seguros de que la subconsulta va a retornar un único resultado.
- Operador IN: se usa cuando la subconsulta va a retornar más de un resultado y buscamos que nuestra consulta sea uno de los valores retornados por la subconsulta.

```
SELECT loc FROM dept WHERE deptno = (SELECT deptno FROM emp WHERE
ename='KING');
```

loc
NEW YORK

```
SELECT loc FROM dept WHERE deptno IN (SELECT deptno FROM emp WHERE
sal>2000);
```

loc
DALLAS
CHICAGO
NEW YORK

5. Alias

Los alias permiten asignar un nombre alternativo a campos, expresiones o a tablas completas. Esto se realiza mediante la palabra clave AS:


```
SELECT loc AS ciudad FROM dept WHERE deptno IN (SELECT deptno FROM emp
WHERE sal>2000);
```

```
+----------+
| ciudad   |
+----------+
| DALLAS   |
| CHICAGO  |
| NEW YORK |
+----------+
```

6. Combinación de tablas con JOIN

El operador JOIN se creó para evitar tener que escribir en la cláusula WHERE las condiciones de relación entre tablas.

Consideremos el siguiente ejemplo:

```
SELECT * FROM emp, dept WHERE emp.deptno=30 AND emp.deptno=dept.deptno;
```

En este caso, una parte de la cláusula WHERE la dedicamos a indicar cómo hacer la unión entre tablas.

Existen diferentes tipos de cláusulas JOIN.

6.1 CROSS JOIN (Unión cruzada).

Consiste en realizar el producto cartesiano de las tablas que componen la unión. El resultado de las dos siguientes consultas es el mismo:

```
SELECT * FROM emp, dept;
SELECT * FROM emp CROSS JOIN dept;
```

empno	ename	job	mgr	hiredate	sal	comm	deptno	deptno	dname	loc
7369	SMITH	CLERK	7902	1980-12-17	800.00	NULL	20	10	ACCOUNTING	NEW YORK
7499	ALLEN	SALESMAN	7698	1981-02-20	1600.00	300.00	30	10	ACCOUNTING	NEW YORK
7521	WARD	SALESMAN	7698	1981-02-22	1250.00	500.00	30	10	ACCOUNTING	NEW YORK
7566	JONES	MANAGER	7839	1981-04-02	2975.00	NULL	20	10	ACCOUNTING	NEW YORK
7654	MARTIN	SALESMAN	7698	1981-09-28	1250.00	1400.00	30	10	ACCOUNTING	NEW YORK
7698	BLAKE	MANAGER	7839	1981-05-01	2850.00	NULL	30	10	ACCOUNTING	NEW YORK
7782	CLARK	MANAGER	7839	1981-06-09	2450.00	NULL	10	10	ACCOUNTING	NEW YORK
7788	SCOTT	ANALYST	7566	1987-07-13	3000.00	NULL	20	10	ACCOUNTING	NEW YORK
7839	KING	PRESIDENT	NULL	1981-11-17	5000.00	NULL	10	10	ACCOUNTING	NEW YORK
7844	TURNER	SALESMAN	7698	1981-09-08	1500.00	0.00	30	10	ACCOUNTING	NEW YORK
7876	ADAMS	CLERK	7788	1987-07-13	1100.00	NULL	20	10	ACCOUNTING	NEW YORK
7900	JAMES	CLERK	7698	1981-12-03	950.00	NULL	30	10	ACCOUNTING	NEW YORK
7902	FORD	ANALYST	7566	1981-12-03	3000.00	NULL	20	10	ACCOUNTING	NEW YORK
7934	MILLER	CLERK	7782	1982-01-23	1300.00	NULL	10	10	ACCOUNTING	NEW YORK
7369	SMITH	CLERK	7902	1980-12-17	800.00	NULL	20	20	RESEARCH	DALLAS
7499	ALLEN	SALESMAN	7698	1981-02-20	1600.00	300.00	30	20	RESEARCH	DALLAS
7521	WARD	SALESMAN	7698	1981-02-22	1250.00	500.00	30	20	RESEARCH	DALLAS
7566	JONES	MANAGER	7839	1981-04-02	2975.00	NULL	20	20	RESEARCH	DALLAS
7654	MARTIN	SALESMAN	7698	1981-09-28	1250.00	1400.00	30	20	RESEARCH	DALLAS
7698	BLAKE	MANAGER	7839	1981-05-01	2850.00	NULL	30	20	RESEARCH	DALLAS
7782	CLARK	MANAGER	7839	1981-06-09	2450.00	NULL	10	20	RESEARCH	DALLAS
7788	SCOTT	ANALYST	7566	1987-07-13	3000.00	NULL	20	20	RESEARCH	DALLAS
7839	KING	PRESIDENT	NULL	1981-11-17	5000.00	NULL	10	20	RESEARCH	DALLAS
7844	TURNER	SALESMAN	7698	1981-09-08	1500.00	0.00	30	20	RESEARCH	DALLAS
7876	ADAMS	CLERK	7788	1987-07-13	1100.00	NULL	20	20	RESEARCH	DALLAS
7900	JAMES	CLERK	7698	1981-12-03	950.00	NULL	30	20	RESEARCH	DALLAS
7902	FORD	ANALYST	7566	1981-12-03	3000.00	NULL	20	20	RESEARCH	DALLAS
7934	MILLER	CLERK	7782	1982-01-23	1300.00	NULL	10	20	RESEARCH	DALLAS
7369	SMITH	CLERK	7902	1980-12-17	800.00	NULL	20	30	SALES	CHICAGO
7499	ALLEN	SALESMAN	7698	1981-02-20	1600.00	300.00	30	30	SALES	CHICAGO
7521	WARD	SALESMAN	7698	1981-02-22	1250.00	500.00	30	30	SALES	CHICAGO
7566	JONES	MANAGER	7839	1981-04-02	2975.00	NULL	20	30	SALES	CHICAGO
7654	MARTIN	SALESMAN	7698	1981-09-28	1250.00	1400.00	30	30	SALES	CHICAGO
7698	BLAKE	MANAGER	7839	1981-05-01	2850.00	NULL	30	30	SALES	CHICAGO
7782	CLARK	MANAGER	7839	1981-06-09	2450.00	NULL	10	30	SALES	CHICAGO
7788	SCOTT	ANALYST	7566	1987-07-13	3000.00	NULL	20	30	SALES	CHICAGO
7839	KING	PRESIDENT	NULL	1981-11-17	5000.00	NULL	10	30	SALES	CHICAGO
7844	TURNER	SALESMAN	7698	1981-09-08	1500.00	0.00	30	30	SALES	CHICAGO
7876	ADAMS	CLERK	7788	1987-07-13	1100.00	NULL	20	30	SALES	CHICAGO
7900	JAMES	CLERK	7698	1981-12-03	950.00	NULL	30	30	SALES	CHICAGO
7902	FORD	ANALYST	7566	1981-12-03	3000.00	NULL	20	30	SALES	CHICAGO
7934	MILLER	CLERK	7782	1982-01-23	1300.00	NULL	10	30	SALES	CHICAGO
7369	SMITH	CLERK	7902	1980-12-17	800.00	NULL	20	40	OPERATIONS	BOSTON
7499	ALLEN	SALESMAN	7698	1981-02-20	1600.00	300.00	30	40	OPERATIONS	BOSTON
7521	WARD	SALESMAN	7698	1981-02-22	1250.00	500.00	30	40	OPERATIONS	BOSTON
7566	JONES	MANAGER	7839	1981-04-02	2975.00	NULL	20	40	OPERATIONS	BOSTON
7654	MARTIN	SALESMAN	7698	1981-09-28	1250.00	1400.00	30	40	OPERATIONS	BOSTON
7698	BLAKE	MANAGER	7839	1981-05-01	2850.00	NULL	30	40	OPERATIONS	BOSTON
7782	CLARK	MANAGER	7839	1981-06-09	2450.00	NULL	10	40	OPERATIONS	BOSTON
7788	SCOTT	ANALYST	7566	1987-07-13	3000.00	NULL	20	40	OPERATIONS	BOSTON
7839	KING	PRESIDENT	NULL	1981-11-17	5000.00	NULL	10	40	OPERATIONS	BOSTON
7844	TURNER	SALESMAN	7698	1981-09-08	1500.00	0.00	30	40	OPERATIONS	BOSTON
7876	ADAMS	CLERK	7788	1987-07-13	1100.00	NULL	20	40	OPERATIONS	BOSTON
7900	JAMES	CLERK	7698	1981-12-03	950.00	NULL	30	40	OPERATIONS	BOSTON
7902	FORD	ANALYST	7566	1981-12-03	3000.00	NULL	20	40	OPERATIONS	BOSTON
7934	MILLER	CLERK	7782	1982-01-23	1300.00	NULL	10	40	OPERATIONS	BOSTON

56 rows in set (0,001 sec)

6.2 JOIN (Unión simple).

Se usa en la cláusula FROM y la sintaxis es:

- **Opción 1**:
  tabla1 [LEFT | RIGHT | FULL OUTER] JOIN tabla2 USING (columnas) [JOIN tabla3 USING (columnas) …]

- **Opción 2**:
  tabla1 [LEFT | RIGHT | FULL OUTER] JOIN tabla2 ON condición

El objetivo es combinar las tablas que se indican en el FROM a través de JOIN. Se pueden añadir tantas como tablas participan en la asociación. Si se usa la opción USING, entre paréntesis se indican las columnas que se deben cruzar, mientras que, si se usa ON, se indica la condición de asociación.

Las siguientes consultas son equivalentes:

```
SELECT ename, job, dname, loc
FROM emp, dept
WHERE emp.deptno=dept.deptno;

SELECT ename, job, dname, loc
FROM emp JOIN dept USING (deptno);

SELECT ename, job, dname, loc
FROM emp JOIN dept ON emp.deptno=dept.deptno;
```

```
+--------+-----------+------------+----------+
| ename  | job       | dname      | loc      |
+--------+-----------+------------+----------+
| SMITH  | CLERK     | RESEARCH   | DALLAS   |
| ALLEN  | SALESMAN  | SALES      | CHICAGO  |
| WARD   | SALESMAN  | SALES      | CHICAGO  |
| JONES  | MANAGER   | RESEARCH   | DALLAS   |
| MARTIN | SALESMAN  | SALES      | CHICAGO  |
| BLAKE  | MANAGER   | SALES      | CHICAGO  |
| CLARK  | MANAGER   | ACCOUNTING | NEW YORK |
| SCOTT  | ANALYST   | RESEARCH   | DALLAS   |
| KING   | PRESIDENT | ACCOUNTING | NEW YORK |
| TURNER | SALESMAN  | SALES      | CHICAGO  |
| ADAMS  | CLERK     | RESEARCH   | DALLAS   |
| JAMES  | CLERK     | SALES      | CHICAGO  |
| FORD   | ANALYST   | RESEARCH   | DALLAS   |
| MILLER | CLERK     | ACCOUNTING | NEW YORK |
+--------+-----------+------------+----------+
```

6.3 NATURAL JOIN (Unión natural).

Cuando entre dos tablas solo existe un campo que se llama igual en ambas y donde uno es clave primaria y en el otro es clave ajena, se puede hacer la operación producto natural. El producto natural es el producto cartesiano y la condición de asociación al mismo tiempo.


```
SELECT * FROM emp NATURAL JOIN dept;
```

deptno	empno	ename	job	mgr	hiredate	sal	comm	dname	loc
20	7369	SMITH	CLERK	7902	1980-12-17	800.00	NULL	RESEARCH	DALLAS
30	7499	ALLEN	SALESMAN	7698	1981-02-20	1600.00	300.00	SALES	CHICAGO
30	7521	WARD	SALESMAN	7698	1981-02-22	1250.00	500.00	SALES	CHICAGO
20	7566	JONES	MANAGER	7839	1981-04-02	2975.00	NULL	RESEARCH	DALLAS
30	7654	MARTIN	SALESMAN	7698	1981-09-28	1250.00	1400.00	SALES	CHICAGO
30	7698	BLAKE	MANAGER	7839	1981-05-01	2850.00	NULL	SALES	CHICAGO
10	7782	CLARK	MANAGER	7839	1981-06-09	2450.00	NULL	ACCOUNTING	NEW YORK
20	7788	SCOTT	ANALYST	7566	1987-07-13	3000.00	NULL	RESEARCH	DALLAS
10	7839	KING	PRESIDENT	NULL	1981-11-17	5000.00	NULL	ACCOUNTING	NEW YORK
30	7844	TURNER	SALESMAN	7698	1981-09-08	1500.00	0.00	SALES	CHICAGO
20	7876	ADAMS	CLERK	7788	1987-07-13	1100.00	NULL	RESEARCH	DALLAS
30	7900	JAMES	CLERK	7698	1981-12-03	950.00	NULL	SALES	CHICAGO
20	7902	FORD	ANALYST	7566	1981-12-03	3000.00	NULL	RESEARCH	DALLAS
10	7934	MILLER	CLERK	7782	1982-01-23	1300.00	NULL	ACCOUNTING	NEW YORK

6.4 Asociaciones internas y externas

El operador JOIN, por defecto, solo cruza los valores iguales entre claves ajenas y claves primarias. Esta operación se denomina asociación interna (INNER).

Una asociación también puede ser externa. En las combinaciones asociaciones externas se combinan todos los registros de una columna, aunque no estén relacionados. Para indicar cuál de las dos columnas se expande se usa LEFT o RIGHT según la columna que nos interese expandir. También podemos usar FULL si queremos expandir ambas columnas

```
SELECT ename, job, dname, loc FROM emp RIGHT JOIN dept USING (deptno);
```

ename	job	dname	loc
SMITH	CLERK	RESEARCH	DALLAS
ALLEN	SALESMAN	SALES	CHICAGO
WARD	SALESMAN	SALES	CHICAGO
JONES	MANAGER	RESEARCH	DALLAS
MARTIN	SALESMAN	SALES	CHICAGO
BLAKE	MANAGER	SALES	CHICAGO
CLARK	MANAGER	ACCOUNTING	NEW YORK
SCOTT	ANALYST	RESEARCH	DALLAS
KING	PRESIDENT	ACCOUNTING	NEW YORK
TURNER	SALESMAN	SALES	CHICAGO
ADAMS	CLERK	RESEARCH	DALLAS
JAMES	CLERK	SALES	CHICAGO
FORD	ANALYST	RESEARCH	DALLAS
MILLER	CLERK	ACCOUNTING	NEW YORK
NULL	NULL	OPERATIONS	BOSTON