Base de Datos
Boletín 4.2: SQL LMD

Para la base de datos *Robótica*, se pide:

1. Obtener el nombre y apellidos de todos los empleados.

```
SELECT nombre, apellidos
FROM Empleados;
```

2. Listar todos los lugares en los que disponemos de algún departamento.

```
SELECT DISTINCTROW lugar
FROM Lugares_dptos;
```

3. Obtener toda la información del empleado con dni 33344555.

```
SELECT *
FROM Empleados
WHERE dni = 33344555;
```

4. Listar los empleados que no tienen un supervisor.

```
SELECT *
FROM Empleados
WHERE dnisuper IS NULL;
```

5. Listar los proyectos cuyo nombre empieza por 'P'.

```
 SELECT *
FROM Proyectos
WHERE nombrep LIKE 'P%';
```

6. Obtener el nombre de todos los dependientes.

```
SELECT nombre
FROM Dependientes;
```

7. Mostrar el dni de todos los empleados que tiene un dependiente a su cargo. Evitar que salgan valores repetidos. Ordenar la consulta.

```
SELECT DISTINCT dniempl
FROM Dependientes
ORDER BY 1;
```

8. Nombre, apellidos y salario de todos los empleados con salario menor de 2000 euros.

```
SELECT nombre, apellidos, salario
FROM Empleados
WHERE salario < 2000;
```

9. Apellidos y nombre de todos los empleados ordenados alfabéticamente.

```
SELECT apellidos, nombre
FROM Empleados
ORDER BY apellidos, nombre;
```

10. Listar los proyectos ordenados alfabéticamente por lugar y nombre del proyecto.

```
SELECT *
FROM Proyectos
ORDER BY lugarp, nombrep;
```

11. Mostrar todos los tratamientos que aplicamos a los empleados: “Sr.”, “Sra.”, etc. Sin que aparezcan repetidos.

```
SELECT DISTINCT tratamiento
FROM Empleados;
```

12. Realizar una consulta donde aparezcan el nombre, apellido y tratamiento con el formato:

    <tratamiento> <apellidos> (de nombre: <nombre>)

    Un ejemplo:

    Sra. Valdés (de nombre: Sonia)

    La cadena anterior debe aparecer en un único campo.

```
SELECT CONCAT(tratamiento,
              '␣' ,
              apellidos,
              '␣(de␣nombre:␣',
              nombre, ')') AS Formato
FROM Empleados;
```

13. Nombre y dirección de los empleados que viven en Sevilla.

```
SELECT nombre, direccion
FROM Empleados
WHERE Empleados.direccion LIKE "%Sevilla%";
```

14. Necesitamos asignar un usuario a cada empleado. Realizar una consulta que muestre junto al nombre de cada empleado, el nombre de usuario. Este estará formado por las primeras tres letras del nombre, las primeras tres letras del apellido y las últimas tres cifras del dni.

```
 SELECT nombre, apellidos, dni,
        CONCAT(LEFT(nombre, 3), LEFT(apellidos, 3), RIGHT(dni,3)) AS Usuario
FROM Empleados;
```

15. Listar todos los empleados por fecha de nacimiento.

```
SELECT *
FROM Empleados
ORDER BY fechanac DESC;
```

16. Listar la información del empleado de mayor edad.

```
SELECT *
FROM Empleados
ORDER BY fechanac
LIMIT 1;
```

17. Igual que el ejercicio anterior pero mostrando también la edad actual.

```
SELECT *, year(now())- year(fechanac) AS edad
FROM Empleados
ORDER BY fechanac
LIMIT 1;
```

18. De los empleados con salario mayor de 2000 euros, obtener el nombre, apellidos y salario. Ordenar la consulta por el sueldo de forma descendente y renombrar la columna salario como Sueldo.

```
SELECT nombre, apellidos, salario AS sueldo
FROM Empleados
WHERE salario>2000;
```

19. Mostrar el nombre de los dependientes que cumplen años en mayo.

```
\begin{lstlisting}
SELECT nombre, fechanac
FROM Dependientes
WHERE MONTH(Dependientes.fechanac) = 5;

SELECT nombre, fechanac
FROM Dependientes
WHERE MONTHNAME(Dependientes.fechanac) = 'may';
```

20. Mostrar el nombre de los dependientes que cumplen años en mayo y en enero. Realizar la consulta utilizando UNION.

```
SELECT nombre, fechanac
FROM Dependientes
WHERE MONTH(Dependientes.fechanac) = 5
UNION
SELECT nombre, fechanac
FROM Dependientes
WHERE MONTH(Dependientes.fechanac) = 1;
```

Hay que recordar que para utilizar `UNION` es necesario que ambas consultas devuelvan el mismo número de campos y que el tipp de los campos de la primera consulta coincidan con el tipo de los campos de la segunda consulta.

21. Listar la información de cada empleado junto a la información del departamento al que pertenece.

```
SELECT *
FROM Empleados INNER JOIN Departamentos
     ON Empleados.numd = Departamentos.numd;
```

De esta forma no saldrian los empleados sin departamentos, para conseguir esto sería:

```
SELECT *
FROM Empleados LEFT JOIN Departamentos
     ON Empleados.numd = Departamentos.numd;
```

22. Cita los empleados que tienen un dependiente de igual nombre que ellos.

```
SELECT dni, Empleados.nombre, apellidos
FROM Empleados INNER JOIN Dependientes
     ON Empleados.dni = Dependientes.dniempl
WHERE Empleados.nombre = Dependientes.nombre;
```

Otra forma de hacerlo es con subconsultas.

```
SELECT nombre
FROM Empleados
WHERE nombre IN (SELECT nombre
                 FROM Dependientes
                 WHERE Dependientes.dniempl = Empleados.dni);
```

La subconsulta devuelve una lista con los nombres de los dependientes para el empleado de la consulta principal. Si el nombre del empleado se encuentra en la lista de los nombres de los dependientes, resulta que empleado y dependiente comparten nombre.

23. Empleados supervisados por Federico Vizcarra.

```
SELECT emp.dni, emp.nombre, emp.apellidos
FROM Empleados as emp INNER JOIN Empleados as jefe
ON emp.dnisuper = jefe.dni
WHERE jefe.nombre = "Federico" AND jefe.apellidos = "Vizcarra";
```

Podemos hacerlo mediante una subconsulta:

```
SELECT dni, nombre
FROM Empleados
WHERE dnisuper IN (SELECT dni
                   FROM Empleados
                   WHERE nombre="Federico" AND
                         apellidos = "Vizcarra");
```

Si tenemos la certeza que solo existe un Federico Vizcarra podemos utilizar el operador = en lugar de IN.

24. Nombres de todos los empleados del departamento 5 que trabajen más de 10 horas en el proyecto ProductoX.

```
SELECT Empleados.dni, Empleados.nombre, Empleados.apellidos
FROM Empleados INNER JOIN Trabaja_en
          ON Empleados.dni = Trabaja_en.dni
     INNER JOIN Proyectos
          ON Trabaja_en.nump = Proyectos.nump
WHERE Empleados.numd = 5 AND
      Trabaja_en.horas > 10 AND
      Proyectos.nombrep = "ProductoX";
```

Con subconsultas:

```
SELECT dni, nombre, apellidos
FROM Empleados
WHERE numd=5 AND
      dni IN (SELECT dni
              FROM Trabaja_en
              WHERE Trabaja_en.horas >= 10 AND
                    Trabaja_en.nump = (SELECT Proyectos.nump
                                       FROM Proyectos
                                       WHERE Proyectos.nombrep="ProductoX"
                                      )
             );

SELECT dni, nombre, apellidos
FROM Empleados
WHERE numd=5 AND
      EXISTS (SELECT *
              FROM Trabaja_en
              WHERE horas >= 10 AND
                    dni = Empleados.dni
                    nump = (SELECT nump FROM Proyectos
                            WHERE Proyectos.nombrep="ProductoX"));
```

En la última subconsulta suponemos que solo existe un proyecto *ProductoX*. En caso contrario hay que utilizar IN.

25. Obtener los empleados femeninos que tienen a su cargo al menos un dependiente.

```
SELECT *
FROM Empleados INNER JOIN Dependientes
     ON Empleados.dni = Dependientes.dniempl
WHERE Empleados.sexo = 'F' ;

SELECT *
FROM Empleados
WHERE sexo ='F' AND
      dni IN (SELECT dniempl FROM Dependientes);
```

    La subconsulta devuelve una lista (de dni's) de todos los empleados que tienen dependientes.

26. Obtener de cada empleado su nombre y el nombre de su supervisor.

```
SELECT emp.nombre AS Empleado, jefe.nombre AS Jefe
FROM Empleados AS emp INNER JOIN Empleados AS jefe
    ON emp.dnisuper = jefe.dni
ORDER BY jefe.nombre;
```

    Ordenamos por jefe para que se vea mejor.

27. Obtener para cada supervisor el número de empleados que supervisa.

```
SELECT dnisuper AS Jefe, COUNT(*)
FROM Empleados
GROUP BY dnisuper;
```

28. Existe algún departamento donde todos los empleados son menores de edad. Como en la tabla *Empleados* disponemos del departamento y de la información para calcular la edad no necesitamos más tablas. Si en un departamento hay alguien que es mayor de edad, ya no se cumple que en el departamento todos sean menores. En lugar de ver la edad de cualquiera del departamento, comprobaremos la edad del empleados de mayor edad. Si el mayor de un departamento tiene, por ejemplo, 30 años; esto significa que en ese departamento no todos son mayores de edad.

    Por otro lado si el mayor de un departamento tiene 17 años, en ese departamento podremos encontrar empleados con 16 años, con 15 años, etc. Si el mayor del departamento es menor de edad, esto significa que el resto de los empleados, en caso de existir, también serán menores.

```
SELECT numd
FROM Empleados
GROUP BY numd
HAVING MAX(YEAR(NOW()) - YEAR(fechanac)) < 18;
```

29. Preparar una lista con los apellidos de todos los gerentes de departamentos que no tienen dependientes.

```
SELECT apellidos
FROM Empleados INNER JOIN Departamentos
     ON Empleados.dni=Departamentos.dnigte
WHERE Empleados.dni NOT IN (SELECT dniempl FROM Dependientes);
```

    Se puede hacer con un `JOIN`, dando prioridad a los empleados. Con esto conseguimos una lista de empleados y sus dependientes serán nulos. Si sus dependientes son nulos, es señal que no tiene dependientes.

```
SELECT apellidos
FROM Empleados INNER JOIN Departamentos
        ON Empleados.dni=Departamentos.dnigte
     LEFT JOIN Dependientes
        ON Empleados.dni = Dependientes.dniempl
WHERE Dependientes.dniempl IS NULL;
```

30. Listar el nombre de los empleados y el nombre de los proyectos en los que trabaja, ordenado por empleados y proyectos.

```
SELECT Empleados.nombre AS Empleado, Proyectos.nombrep AS Proyecto
FROM Empleados INNER JOIN Trabaja_en
          ON Empleados.dni = Trabaja_en.dni)
     INNER JOIN Proyectos
          ON Trabaja_en.nump = Proyectos.nump
ORDER BY Empleados.nombre, Proyectos.nombrep;
```

Hemos de hacer un `JOIN` de *Empleados* con *Proyectos*. El problema es que no hay forma de hacerlo directamente, necesitamos hacer el `JOIN` a través de *Trabaja_en*.

31. Nombres y direcciones de todos los empleados que trabajan en, por lo menos, un proyecto situado en Sevilla pero cuyo departamento (del empleado) no está ubicado ahí.

```
SELECT DISTINCT Empleados.nombre, Empleados.direccion
FROM (Trabaja_en INNER JOIN Proyectos ON Trabaja_en.nump = Proyectos.nump)
              INNER JOIN Empleados ON Empleados.dni = Trabaja_en.dni
WHERE lugarp = "Sevilla" AND
      Empleados.numd IN (SELECT numd
                         FROM Lugares_dptos
                         WHERE Lugares_dptos.lugar <> "Sevilla");
```

¿Por qué falla?

Veamos como se hace sin subconsultas

```
SELECT DISTINCT Empleados.nombre, Empleados.direccion
FROM Trabaja_en INNER JOIN Proyectos
          ON Trabaja_en.nump = Proyectos.nump
     INNER JOIN Empleados
          ON Empleados.dni = Trabaja_en.dni
     INNER JOIN Departamentos
          ON Departamentos.numd = Empleados.numd
     INNER JOIN Lugares_dptos
          ON Lugares_dptos.numd = Departamentos.numd
WHERE lugarp = "Sevilla" AND
      Empleados.numd IN (SELECT numd
                         FROM Lugares_dptos
                         WHERE Lugares_dptos.lugar <> "Sevilla");
```

32. Para los departamentos que se encuentran ubicados en Sevilla, nombre del departamento junto con el nombre del gerente.

```
SELECT Empleados.nombre, Departamentos.nombred
FROM (Empleados INNER JOIN Departamentos
      ON Empleados.dni = Departamentos.dnigte)
     INNER JOIN Lugares_dptos
     ON Departamentos.numd = Lugares_dptos.numd
WHERE Lugares_dptos.lugar = "Sevilla";
```

33. Mostrar los departamentos que actualmente no se encargan de ningún proyecto.

```
SELECT *
FROM Departamentos
WHERE numd NOT IN (SELECT numd FROM Proyectos);
```

La subconsulta nos proporciona una lista con el número de todos los departamentos que son responsables de algún proyecto.

34. Para cada departamento obtener el salario medio de sus empleados.

```
SELECT numd, avg (salario) AS salario_medio
FROM Empleados
GROUP BY numd;
```

35. Igual que el ejercicio anterior pero añadiendo al salario medio “euros”.

```
SELECT numd, concat(FLOOR(avg(salario)),' euros') AS salario_medio
FROM Empleados
GROUP BY numd;

SELECT numd, concat(CONVERT(avg(salario), CHAR) , " euros") AS salario_medio
FROM Empleados
GROUP BY numd;
```

    La función `FLOOR` de MySQL redondea un número con decimales. Y la función `CONVERT` convierte una expresión a un tipo dado (Ver ambas funciones en la documentación de MySQL).

36. Para cada departamento mostrar su número y cuantos empleados trabajan en él.

```
SELECT numd, COUNT(*) AS "Número de empleados"
FROM Empleados
GROUP BY numd;
```

    Agrupamos por departamento y contamos cuantos registros (empleados) existen en cada uno. Toda la información se extrae de la tabla *Empleados*.

37. Mostrar el departamento que tiene el menor número de empleados. En caso de existir más de un departamento con el número mínimo de empleados, mostrarlos todos. Para ver cuantos empleados trabajan en cada departamento:

```
SELECT numd, COUNT(*)
FROM Empleados
GROUP BY numd
ORDER BY 2;
```

    Solo hemos de mostrar los departamentos cuyo número de empleados es mínimo.

```
SELECT numd, COUNT(*)
FROM Empleados
GROUP BY numd
HAVING COUNT(*) = (SELECT COUNT(*)
                   FROM Empleados
                   GROUP BY numd
                   LIMIT 1);
```

    La subconsulta nos devuelve el número mínimo de empleados en un departamento. La consulta principal muestra los departamentos cuyo número de empleados coincide con el devuelto por la subconsulta.

38. Para cada proyecto mostrar cuantas personas trabajan en ellos.

```
SELECT nump, count (dni) AS trabajadores
FROM Trabaja_en
GROUP BY nump;
```

    Otra forma alternativa:

```
SELECT nombrep, COUNT(dni) AS trabajadores
FROM Trabaja_en INNER JOIN Proyectos
ON Trabaja_en.nump = Proyectos.nump
GROUP BY nombrep;
```

`COUNT(dniempl)` es lo mismo que `COUNT(*)`.

39. Para cada departamento obtener el número (sólo el número) de empleados que lo forman, así como el salario medio de estos.

```
SELECT numd, count(dni) AS 'Número de empleados',
       AVG(salario) AS 'Sueldo medio'
FROM Empleados
GROUP BY numd;
```

40. Para cada proyecto, número de empleados que trabajan en él y número de horas total de trabajo.

```
SELECT nombrep, COUNT(dni) AS trabajadores, SUM(horas) AS Total_horas
FROM Trabaja_en INNER JOIN Proyectos
     ON Trabaja_en.nump = Proyectos.nump
GROUP BY nombrep;
```

41. Para cada departamento, salario medio de sus empleados y número de empleados.

```
SELECT Departamentos.numd, Departamentos.nombred,
       COUNT(*) AS 'Número de empleados', AVG(salario) AS 'Sueldo medio'
FROM Empleados INNER JOIN Departamentos
     ON Empleados.numd = Departamentos.numd
GROUP BY Departamentos.numd, Departamentos.nombred;
```

42. Para cada empleado obtener su nombre y el número de personas que dependen de él.

```
SELECT Empleados.nombre, COUNT(*) AS Dependientes
FROM Empleados INNER JOIN Dependientes
     ON Empleados.dni = Dependientes.dniempl
GROUP BY Empleados.nombre;
```

43. Obtener cuantos hombres y mujeres trabajan en la empresa.

```
SELECT sexo, COUNT(dni) AS Empleados
FROM Empleados
GROUP BY sexo;
```

44. Para cada departamento obtener el sueldo máximo y mínimo de los empleados.

```
SELECT numd, MAX(salario) AS máximo, MIN(salario) AS mínimo
FROM Empleados
GROUP BY numd;
```

45. Listar los empleados que pertenecen al departamento 4 y ganan más de 1500 euros, y los empleados del departamento 5 que ganan más de 2000 euros.

```
SELECT *
FROM Empleados
WHERE (numd=4 AND salario>25000) OR
      (numd=5 AND salario<30000);
```

46. Obtener los dni, de todos los empleados que trabajan en el departamento 5 y de todos los jefes. Utilizar UNION.

```
SELECT dni
FROM Empleados
WHERE numd=5
UNION
SELECT dnisuper AS dni
FROM Empleados;
```

47. Suponiendo que la jubilación se produce a los 65 años. Calcular cuántos años le quedan a cada empleados para jubilarse. Mostrar el nombre y edad de cada empleado.

```
SELECT 65 - (year(now())-year(fechanac)) AS 'Para␣jubilación',
       year(now())-year(fechanac) AS Edad, nombre
FROM Empleados;
```

48. Mostrar un informe con el nombre y el salario de todos los empleados. Si el empleado en cuestión es un gerente de departamento anteponer a su nombre "GERENTE: " y en caso contrario anteponer al nombre "EMPLEADO: ".

```
SELECT CONCAT("GERENTE:␣",nombre) AS 'Tipo␣Empleado', apellidos
FROM Departamentos JOIN Empleados
  ON Departamentos.dnigte = Empleados.dni
UNION
SELECT CONCAT("EMPLEADO:␣", nombre) AS 'Tipo␣Empleado', apellidos
FROM Empleados
WHERE dni NOT IN (SELECT dnigte FROM Departamentos);
```

49. Obtener nombre, apellidos y fecha de nacimiento de todos los empleados con salario inferior a 2000, ordenados alfabéticamente por el apellido.

```
SELECT nombre, apellidos, fechanac
FROM Empleados
WHERE salario < 2000
ORDER BY apellidos;
```

50. Obtener el salario medio de los empleadoss de sexo femenino.

```
SELECT AVG(salario) AS salario_medio
FROM Empleados
WHERE sexo = 'F';
```

51. Para cada proyecto, mostrar el nombre del proyecto y el total de horas que trabajan todos los empleados en él.

```
SELECT nombrep, SUM(horas) as total_horas
FROM Proyectos INNER JOIN Trabaja_en
     ON Proyectos.nump = Trabaja_en.nump
GROUP BY nombrep;
```

52. Empleados que no trabajan en ningún proyecto.

```
SELECT dni, nombre, apellidos
FROM Empleados
WHERE dni NOT IN (SELECT dni FROM Trabaja_en)
```

    La subconsulta nos lista los dni de los empleados que trabajan en algún proyecto. Con lo cual nos interesan lo empleados que no aparecen en esta lista.

53. Realizar una consulta que muestre el nombre de cada proyecto, el número y nombre del departamento responsable y el nombre del gerente de dicho departamento. Ordenar todo por número de departamento

```
SELECT nombrep AS Proyecto, Departamentos.numd AS 'Num.␣depart.',
       nombred AS Departamento, nombre AS Gerente,
       apellidos AS 'Apellidos␣gerente'
FROM (Proyectos JOIN Departamentos
      ON Proyectos.numd = Departamentos.numd)
      JOIN Empleados ON Departamentos.dnigte = Empleados.dni
ORDER BY Departamentos.numd
```

54. Para cada empleado mostrar su dni, nombre y horas total de trabajo, así como el número de proyectos en los que participa.

```
SELECT Empleados.dni, Empleados.nombre,
       SUM(horas) AS Horas, COUNT(*) AS Proyectos
FROM Trabaja_en RIGHT JOIN Empleados
     ON Trabaja_en.dni = Empleados.dni
GROUP BY Empleados.dni, Empleados.nombre
```

55. Determinar para cada empleado cuantas horas trabaja en total en los distintos proyectos.

```
SELECT dni, SUM(horas) AS Horas
FROM Trabaja_en
GROUP BY dni
```

56. Para cada proyecto, mostrar el nombre, número y horas de dedicación total (de todos los empleados que trabajan en él).

```
SELECT Proyectos.nump, Proyectos.nombrep, SUM(horas) AS "Total␣horas"
FROM Proyectos JOIN Trabaja_en
     ON Proyectos.nump = Trabaja_en.nump
GROUP BY Proyectos.nump, Proyectos.nombrep;
```

57. Listar los datos de los empleados que no trabajan en ningún proyecto ubicado en “Camas”.

```
SELECT *
FROM Empleados
WHERE NOT EXISTS (SELECT *
                  FROM Proyectos JOIN Trabaja_en
                  USING(nump)
                  WHERE lugarp = "Camas" AND
                        Trabaja_en.dni = Empleados.dni);
```