Ejercico 1

-- 1. Lista el apellido de todos los empleados.

SELECT apellido FROM empleados;

-- 2. Lista el apellido de los empleados eliminando los apellidos que estén repetidos.

SELECT DISTINCT apellido FROM empleados;

-- 3. Lista todas las columnas de la tabla empleados.

SELECT * FROM empleados;

-- 4. Lista el nombre y apellido de todos los empleados.

SELECT nombre, apellido FROM empleados ;

-- 5. Lista el cuit/cuil de los departamentos de los empleados que aparecen en la tabla empleados.

SELECT cuil_cuit FROM empleados INNER JOIN departamentos WHERE id_departamento=iddepartamento ;

-- 6. Lista el nombre y apellido de los empleados en una única columna.

SELECT CONCAT_WS(' ', nombre,apellido) AS nombre_apellido FROM empleados ;

-- 7. Lista el nombre y apellido de los empleados en una única columna, convirtiendo todos los caracteres en mayúscula.

SELECT UPPER(CONCAT_WS(' ', nombre,apellido)) AS nombre_apellido FROM empleados ;

-- Lista el nombre y apellido de los empleados en una única columna, convirtiendo todos los caracteres en minúscula.

SELECT LOWER(CONCAT_WS(' ', nombre,apellido)) AS nombre_apellido FROM empleados ;

-- 9. Lista el nombre de los departamentos y el valor del presupuesto actual ordenado de forma ascendente.

SELECT nombre, presupuesto FROM departamentos ORDER BY presupuesto ASC;

-- 10. Lista el nombre de todos los departamentos ordenados de forma ascendente.

SELECT nombre FROM departamentos ORDER BY nombre ASC;

-- 11. Lista el nombre de todos los departamentos ordenados de forma descendente.

SELECT nombre FROM departamentos ORDER BY nombre DESC;

-- 12. Lista el apellido y el nombre de todos los empleados, ordenados de forma alfabética tendiendo en cuenta en primer lugar su apellido y luego su nombre.

SELECT nombre,apellido FROM empleados ORDER BY apellido , nombre asc;

-- 13. Devuelve una lista con el nombre y el presupuesto, de los 3 departamentos que tienen mayor presupuesto.

SELECT nombre, presupuesto FROM departamentos ORDER BY presupuesto DESC LIMIT 3;

-- 14. Devuelve una lista con el nombre y el presupuesto, de los 3 departamentos que tienen menor presupuesto.

SELECT nombre, presupuesto FROM departamentos ORDER BY presupuesto ASC LIMIT 3;

-- 15. Devuelve una lista con el nombre de los departamentos y el presupuesto, de aquellos que tienen un presupuesto mayor o igual a $150000.

SELECT nombre, presupuesto FROM departamentos WHERE presupuesto > 150000;

-- 16. Devuelve una lista con el nombre de los departamentos y el presupuesto, de aquellos que tienen un presupuesto entre $100000 y $200000. Sin utilizar el operador BETWEEN.

SELECT nombre, presupuesto FROM departamentos WHERE presupuesto > 100000 AND presupuesto < 200000;

-- 17. Devuelve una lista con el nombre de los departamentos que no tienen un presupuesto entre $100000 y $200000. Sin utilizar el operador BETWEEN.

SELECT nombre, presupuesto FROM departamentos WHERE presupuesto < 100000 AND presupuesto > 200000;

-- 18. Devuelve una lista con el nombre de los departamentos que tienen un presupuesto entre $100000 y $200000. Utilizando el operador BETWEEN.

SELECT nombre, presupuesto FROM departamentos WHERE presupuesto BETWEEN 100000 AND 200000;

-- 19. Devuelve una lista con el nombre de los departamentos que no tienen un presupuesto entre $100000 y $200000. Utilizando el operador BETWEEN.

SELECT nombre, presupuesto FROM departamentos WHERE presupuesto NOT BETWEEN 100000 AND 200000;

-- 20. Devuelve un listado con los empleados y los datos de los departamentos donde trabaja cada uno.

SELECT empleados.nombre,empleados.apellido,departamentos.nombre AS departamento,departamentos.presupuesto FROM departamentos INNER JOIN empleados WHERE iddepartamento=id_departamento ;

-- 21. Devuelve un listado con los empleados y los datos de los departamentos donde trabaja cada uno. Ordena el resultado, en primer lugar por el nombre del departamento (en orden alfabético) y en segundo lugar por apellido y el nombre de los empleados.

SELECT empleados.nombre,empleados.apellido,departamentos.nombre AS departamento,departamentos.presupuesto FROM departamentos INNER JOIN empleados WHERE iddepartamento=id_departamento ORDER BY departamentos.nombre,empleados.nombre;

-- 22. Devuelve un listado con el código y el nombre del departamento, solamente de aquellos departamentos que tienen empleados.

SELECT DISTINCT departamentos.iddepartamento, departamentos.nombre AS departamento FROM departamentos INNER JOIN empleados WHERE iddepartamento=id_departamento;

-- 23. Devuelve el nombre del departamento donde trabaja el empleado que tiene el cuit 27-38382980-3.

SELECT departamentos.nombre AS departamento FROM departamentos INNER JOIN empleados WHERE empleados.cuil_cuit="27-38382980-3" AND iddepartamento=id_departamento;

-- 24. Devuelve el nombre del departamento donde trabaja el empleado Pepe Ruiz.

SELECT departamentos.nombre AS departamento FROM departamentos INNER JOIN empleados WHERE empleados.nombre="Pepe" AND empleados.apellido="Ruiz" AND iddepartamento=id_departamento;

-- 25. Devuelve un listado con los datos de los empleados que trabajan en el departamento de I+D. Ordena el resultado alfabéticamente.

SELECT empleados.nombre, empleados.apellido FROM departamentos INNER JOIN empleados WHERE departamentos.nombre ="I+D" AND iddepartamento=id_departamento ORDER BY empleados.nombre ASC;

-- 26. Devuelve un listado con los datos de los empleados que trabajan en el departamento de Sistemas, Contabilidad o I+D. Ordena el resultado alfabéticamente.

SELECT DISTINCT empleados.nombre, empleados.apellido FROM empleados WHERE id_departamento ="2" or id_departamento ="4" or id_departamento ="5" order BY nombre ASC;

-- 27. Devuelve una lista con el nombre de los empleados que tienen los departamentos que no tienen un presupuesto entre $100000 y $200000.

SELECT empleados.nombre FROM empleados INNER JOIN departamentos WHERE departamentos.presupuesto NOT BETWEEN 100000 AND 200000 and iddepartamento=id_departamento;

Ejercicio 2

-- 1. Devuelve un listado con todos los pedidos que se han realizado. Los pedidos deben estar ordenados por la fecha de realización, mostrando en primer lugar los pedidos más recientes.

SELECT * from pedidos ORDER BY fecha DESC;

-- 2. Devuelve todos los datos de los dos pedidos de mayor valor.

SELECT * from pedidos ORDER BY cantidad DESC LIMIT 2 ;

-- 3. Devuelve un listado con los identificadores de los clientes que han realizado algún pedido. Tenga en cuenta que no debe mostrar identificadores que estén repetidos.

SELECT DISTINCT id_cliente from clientes INNER JOIN pedidos WHERE pedidos.id_cliente=clientes.id ;

-- 4. Devuelve un listado de todos los pedidos que se realizaron durante el año 2022, cuya cantidad total sea superior a $500.

SELECT * FROM pedidos WHERE fecha > "2022/01/01" AND fecha <= "2022/12/31" AND cantidad>"500";

-- 5. Devuelve un listado con el nombre y apellido de los vendedores que tienen una comisión entre 0.05 y 0.11.

SELECT nombre, apellido FROM vendedores WHERE comisión BETWEEN "0.05" AND "0.11";

-- 6. Devuelve el valor de la comisión de mayor valor que existe en la tabla vendedores.

SELECT comisión FROM vendedores ORDER BY comisión DESC LIMIT 1;

-- 7. Devuelve el identificador, nombre y primer apellido de aquellos clientes cuyo cuitcuil no es NULL. El listado deberá estar ordenado alfabéticamente por apellido y nombre.

SELECT id, nombre, apellido FROM clientes WHERE NOT cuitcuil="null";

-- Devuelve un listado de los nombres de los clientes que empiezan por “A” y terminan por “n” y también los nombres que empiezan por “P”. El listado deberá estar ordenado alfabéticamente.

SELECT nombre FROM clientes WHERE nombre LIKE "A%" AND nombre LIKE "%n" OR nombre LIKE "P%" ORDER BY nombre ASC;

-- 9. Devuelve un listado de los nombres de los clientes que no empiezan por “A”. El listado deberá estar ordenado alfabéticamente.

SELECT nombre FROM clientes WHERE nombre NOT LIKE "A%" ORDER BY nombre ASC;

-- 10. Devuelve un listado con los nombres de los vendedores que terminan por “el” o “o”. Tenga en cuenta que se deberán eliminar los nombres repetidos.

SELECT DISTINCT nombre FROM vendedores WHERE nombre LIKE "%el" OR nombre LIKE "%O" ORDER BY nombre ASC;

-- 11. Devuelve un listado con el identificador, nombre y apellido de todos los clientes que han realizado algún pedido. El listado debe estar ordenado alfabéticamente y se deben eliminar los elementos repetidos.

SELECT DISTINCT clientes.id, clientes.nombre, clientes.apellido FROM clientes INNER JOIN pedidos WHERE clientes.id=pedidos.id_cliente ORDER BY clientes.nombre ASC;

-- 12. Devuelve un listado que muestre todos los pedidos que ha realizado cada cliente. El resultado debe mostrar todos los datos de los pedidos y del cliente. El listado debe mostrar los datos de los clientes ordenados alfabéticamente.

SELECT * FROM clientes INNER JOIN pedidos WHERE clientes.id=pedidos.id_cliente ORDER BY clientes.nombre ASC;

-- 13. Devuelve un listado que muestre todos los pedidos en los que ha participado un vendedor. El resultado debe mostrar todos los datos de los pedidos y de los vendedores. El listado debe mostrar los datos de los vendedores ordenados alfabéticamente.

SELECT * FROM vendedores INNER JOIN pedidos WHERE vendedores.id=pedidos.id_vendedor ORDER BY vendedores.nombre ASC;

-- 14. Devuelve un listado que muestre todos los clientes, con todos los pedidos que han realizado y con los datos de los vendedores asociados a cada pedido.

SELECT * FROM clientes INNER JOIN pedidos ON pedidos.id_cliente=clientes.id INNER JOIN vendedores ON pedidos.id_vendedor=vendedores.id ORDER BY clientes.nombre ASC;

-- 15. Devuelve un listado de todos los clientes que realizaron un pedido durante el año 2022, cuyo monto esté entre $300 y $1000.

SELECT nombre,apellido FROM clientes INNER JOIN pedidos ON pedidos.id_cliente=clientes.id WHERE pedidos.fecha>31/12/2021 AND pedidos.fecha<01/01/2023 AND pedidos.cantidad BETWEEN 300 AND 1000 ;

-- 16. Devuelve el nombre y apellido de todos los vendedores que han participado en algún pedido realizado por María Santana.

SELECT DISTINCT vendedores.nombre,vendedores.apellido FROM vendedores INNER JOIN pedidos ON vendedores.id=pedidos.id_vendedor INNER JOIN clientes ON pedidos.id_cliente=clientes.id WHERE clientes.nombre='Maria' AND clientes.apellido='Santana';

-- 17. Devuelve el nombre de todos los clientes que han realizado algún pedido con el vendedor Daniel Sáez.

SELECT DISTINCT clientes.nombre FROM clientes INNER JOIN pedidos ON clientes.id=pedidos.id_cliente INNER JOIN vendedores ON pedidos.id_vendedor=vendedores.id WHERE vendedores.nombre='Daniel' AND vendedores.apellido='Sáez';

-- 18. Devuelve un listado con todos los clientes junto con los datos de los pedidos que han realizado. Este listado también debe incluir los clientes que no han realizado ningún pedido. El listado debe estar ordenado alfabéticamente por el apellido y nombre de los clientes.

SELECT * FROM clientes LEFT JOIN pedidos ON clientes.id=pedidos.id_cliente ORDER BY clientes.apellido , clientes.nombre ASC;

-- 19. Devuelve un listado con todos los vendedores junto con los datos de los pedidos que han realizado. Este listado también debe incluir los vendedores que no han realizado ningún pedido. El listado debe estar ordenado alfabéticamente por el apellido y nombre de los vendedores.

SELECT * FROM vendedores LEFT JOIN pedidos ON vendedores.id=pedidos.id_vendedor ORDER BY vendedores.apellido , vendedores.nombre ASC;

-- 20. Devuelve un listado que solamente muestre los clientes que no han realizado ningún pedido.

SELECT * FROM pedidos RIGHT JOIN clientes ON pedidos.id_cliente=clientes.id where pedidos.id_cliente is null ;

-- 21. Devuelve un listado que solamente muestre los vendedores que no han realizado ningún pedido.

SELECT * FROM pedidos RIGHT JOIN vendedores ON pedidos.id_vendedor=vendedores.id where pedidos.id_vendedor is null ;

-- 22. Devuelve un listado con los clientes que no han realizado ningún pedido y de los vendedores que no han participado en ningún pedido. Ordene el listado alfabéticamente por el apellido y el nombre. En el listado deberá diferenciar de algún modo los clientes y los vendedores.

SELECT vendedores.nombre,vendedores.apellido, vendedores.id as vendedor_id,'' as cliente_id FROM pedidos RIGHT JOIN vendedores ON pedidos.id_vendedor=vendedores.id where pedidos.id_vendedor is null UNION SELECT clientes.nombre,clientes.apellido,'',clientes.id as cliente_id FROM pedidos RIGHT JOIN clientes ON pedidos.id_cliente=clientes.id WHERE pedidos.id_cliente IS NULL ORDER BY apellido,nombre;

-- 23. Calcula la cantidad total que suman todos los pedidos que aparecen en la tabla pedido.

SELECT SUM(pedidos.cantidad) AS suma_cantidad FROM pedidos;

-- 24. Calcula la cantidad media de todos los pedidos que aparecen en la tabla pedido.

SELECT AVG(pedidos.cantidad) AS promedio_cantidad FROM pedidos;

-- 25. Calcula el número total de vendedores distintos que aparecen en la tabla pedido.

SELECT DISTINCT COUNT(vendedores.nombre) AS cantidad_de_vendedores FROM vendedores;

-- 26. Calcula el número total de clientes que aparecen en la tabla cliente.

SELECT COUNT(clientes.id) AS cantidad_de_clientes FROM clientes;

-- 27. Calcula cuál es la mayor cantidad que aparece en la tabla pedido.

SELECT MAX(pedidos.cantidad) AS Maxima_cantidad FROM pedidos;

-- 28. Calcula cuál es la menor cantidad que aparece en la tabla pedido.

SELECT MIN(pedidos.cantidad) AS Minima_cantidad FROM pedidos;

-- 29. Calcula cuál es el valor máximo de categoría para cada una de las ciudades que aparece en la tabla cliente.

SELECT clientes.ciudad, MAX(clientes.categoría) AS maxima_categoria FROM clientes group by ciudad;

-- 30. Calcula cuál es el máximo valor de los pedidos realizados durante el mismo día para cada uno de los clientes. Es decir, el mismo cliente puede haber realizado varios pedidos de diferentes cantidades el mismo día. Se pide que se calcule cuál es el pedido de máximo valor para cada uno de los días en los que un cliente ha realizado un pedido. Muestra el identificador del cliente, nombre, apellido, la fecha y el valor de la cantidad.

SELECT clientes.id, clientes.nombre, clientes.apellido,pedidos.fecha,MAX(pedidos.cantidad) FROM clientes INNER JOIN pedidos WHERE clientes.id=pedidos.id_cliente group by pedidos.fecha ORDER BY pedidos.fecha;

-- 31. Calcula cuál es el máximo valor de los pedidos realizados durante el mismo día para cada uno de los clientes, teniendo en cuenta que sólo queremos mostrar aquellos pedidos que superen la cantidad de $2000 .

SELECT clientes.id, clientes.nombre, clientes.apellido,pedidos.fecha,MAX(pedidos.cantidad) FROM clientes INNER JOIN pedidos WHERE clientes.id=pedidos.id_cliente and pedidos.cantidad>2000 group by pedidos.fecha ORDER BY pedidos.fecha;

-- 32. Calcula el máximo valor de los pedidos realizados para cada uno de los vendedores durante la fecha 2021-08-17. Muestra el identificador del vendedor, nombre, apellido y total.

SELECT vendedores.id, vendedores.nombre, vendedores.apellido,max(pedidos.cantidad) FROM vendedores INNER JOIN pedidos WHERE vendedores.id=pedidos.id_vendedor and pedidos.fecha='2021-08-17'GROUP BY vendedores.nombre ;

-- 33. Devuelve un listado con el identificador de cliente, nombre y apellido y el número total de pedidos que ha realizado cada uno de los clientes. Tenga en cuenta que pueden existir clientes que no han realizado ningún pedido. Estos clientes también deben aparecer en el listado indicando que el número de pedidos realizados es 0.

SELECT clientes.id, clientes.nombre, clientes.apellido,count(pedidos.cantidad) FROM clientes LEFT JOIN pedidos ON clientes.id=pedidos.id_cliente GROUP BY clientes.nombre;

-- 34. Devuelve un listado con el identificador de cliente, nombre, apellido y el número total de pedidos que ha realizado cada uno de clientes durante el año 2020.

SELECT clientes.id, clientes.nombre, clientes.apellido,count(pedidos.cantidad) FROM clientes INNER JOIN pedidos WHERE clientes.id=pedidos.id_cliente AND pedidos.fecha>'2019-12-31' AND pedidos.fecha<'2021-01-01' GROUP BY clientes.nombre;

-- 35. Devuelve cuál ha sido el pedido de máximo valor que se ha realizado cada año.

SELECT pedidos.fecha, max(pedidos.cantidad) FROM pedidos WHERE pedidos.fecha like '2018%'

UNION SELECT pedidos.fecha, max(pedidos.cantidad) FROM pedidos WHERE pedidos.fecha like '2019%'

UNION SELECT pedidos.fecha, max(pedidos.cantidad) FROM pedidos WHERE pedidos.fecha like '2020%'

UNION SELECT pedidos.fecha, max(pedidos.cantidad) FROM pedidos WHERE pedidos.fecha like '2021%'

UNION SELECT pedidos.fecha, max(pedidos.cantidad) FROM pedidos WHERE pedidos.fecha like '2022%';

-- 36. Devuelve el número total de pedidos que se han realizado cada año.

SELECT pedidos.fecha, COUNT(pedidos.cantidad) FROM pedidos WHERE pedidos.fecha like '2018%'

UNION SELECT pedidos.fecha, COUNT(pedidos.cantidad) FROM pedidos WHERE pedidos.fecha like '2019%'

UNION SELECT pedidos.fecha, COUNT(pedidos.cantidad) FROM pedidos WHERE pedidos.fecha like '2020%'

UNION SELECT pedidos.fecha, COUNT(pedidos.cantidad) FROM pedidos WHERE pedidos.fecha like '2021%'

UNION SELECT pedidos.fecha, COUNT(pedidos.cantidad) FROM pedidos WHERE pedidos.fecha like '2022%';

-- 37. Devuelve un listado con todos los pedidos que ha realizado Adela Salas. (Sin utilizar INNER JOIN).

SELECT * FROM pedidos CROSS JOIN clientes WHERE pedidos.id_cliente=clientes.id and clientes.nombre='Adela' AND clientes.apellido='Salas';

-- 38. Devuelve el número de pedidos en los que ha participado el vendedor Daniel Sáe. (Sin utilizar INNER JOIN)

SELECT COUNT(pedidos.id) AS cantidad_pedidos FROM pedidos CROSS JOIN vendedores WHERE pedidos.id_vendedor=vendedores.id AND vendedores.nombre='Daniel' AND vendedores.apellido='Sáez';

-- 39. Devuelve los datos del cliente que realizó el pedido más caro en el año 2020. (Sin utilizar INNER JOIN)

SELECT clientes.id,clientes.nombre,clientes.apellido,max(pedidos.cantidad) FROM clientes CROSS JOIN pedidos WHERE clientes.id=pedidos.id_cliente AND pedidos.fecha like '2020%';

-- 40. Devuelve la fecha y la cantidad del pedido de menor valor realizado por el cliente Pepe Ruiz Santana.

SELECT pedidos.fecha,min(pedidos.cantidad) FROM clientes CROSS JOIN pedidos WHERE clientes.id=pedidos.id_cliente AND clientes.nombre='Pepe' AND clientes.apellido='Ruiz';

-- 41. Devuelve un listado de los clientes que no han realizado ningún pedido. (Utilizando IN o NOT IN).

SELECT clientes.nombre,clientes.apellido FROM clientes LEFT JOIN pedidos ON clientes.id=pedidos.id_cliente WHERE COALESCE(pedidos.id_cliente,'') IN ('') ;

-- 42. Devuelve un listado de los vendedores que no han realizado ningún pedido. (Utilizando IN o NOT IN).

SELECT vendedores.nombre,vendedores.apellido FROM vendedores LEFT JOIN pedidos ON vendedores.id=pedidos.id_vendedor WHERE COALESCE(pedidos.id_vendedor,'') IN ('');

-- 43. Devuelve un listado de los clientes que no han realizado ningún pedido. (Utilizando EXISTS o NOT EXISTS).

SELECT clientes.nombre, clientes.apellido FROM clientes WHERE NOT EXISTS (SELECT * From pedidos where clientes.id=pedidos.id_cliente);

-- 44. Devuelve un listado de los vendedores que no han realizado ningún pedido. (Utilizando EXISTS o NOT EXISTS).

SELECT vendedores.nombre, vendedores.apellido FROM vendedores WHERE NOT EXISTS (SELECT * From pedidos where vendedores.id=pedidos.id_vendedor);