**Nivel Principiante**

1. Inserta 5 clientes en la tabla clientes.

insert into Clientes (nombre,correo,pais) values

('Esteban','Esteban@gmail.com','Chile'),

('Marcos','Marcos@gmail.com','Bolivia'),

('Juan','Juan@gmail.com','Argentina'),

('Juan','Juan@gmail.com','Argentina'),

('Ricardo','Ricardo@gmail.com','México');

1. Inserta 5 productos en la tabla productos.

insert into productos (nombre,precio,stock) values

('Platos',100.0,30),

('Lavadora',50.0,90),

('Plancha',110.0,20),

('Reloj',500.0,10),

('Celular',300.0,5);

1. Inserta 5 ventas distintas en la tabla ventas.

insert into ventas (id\_cliente,id\_producto,fecha\_venta,cantidad) values

(1,2,'2020-06-10',20),

(1,2,'2021-01-11',50),

(2,3,'2022-02-20',30),

(3,1,'2023-04-15',40),

(5,4,'2021-08-12',45);

1. Inserta 5 empleados en la tabla empleados.

insert into empleados (nombre,departamento,salario,fecha\_ingreso) values

('Edgar','Cosmeticos',550.0,'2022-02-04'),

('Jhon','Informatica',450.0,'2015-01-04'),

('Rick','Informatica',750.0,'2019-08-14'),

('Erick','Ropa',250.0,'2017-09-22'),

('Vincent','Zapatillas',150.0,'2005-10-21');

1. Muestra todos los clientes.

select \* from clientes;

1. Muestra todos los productos cuyo precio sea mayor a 50.

select \* from productos

where precio > 50;

1. Muestra todos los empleados que trabajan en el departamento "Ventas".

select \* from empleados

where departamento = 'Ventas';

1. Muestra todos los productos con stock menor a 10.

select \* from productos

where stock < 10;

1. Actualiza el correo de un cliente.

update clientes

set correo = 'Esteban1@gmail.com'

where correo = 'Esteban@gmail.com';

1. Borra un cliente que tenga id\_cliente = 3.

delete from clientes

where id\_cliente = 3;

**Nivel Intermedio**

1. Muestra el nombre y correo de todos los clientes de "México".

select nombre, correo from clientes

where pais = "México";

1. Muestra los productos cuyo nombre comience con la letra "T".

select nombre from productos

where nombre like 't%';

1. Muestra las ventas realizadas en 2023.

select \* from ventas

where fecha\_venta between '2023-01-01' and '2023-12-31';

1. Muestra los nombres de clientes que realizaron una venta (JOIN entre clientes y ventas).

select c.nombre, v.id\_venta from ventas v

join clientes c

on v.id\_cliente = c.id\_cliente;

1. Muestra los productos que **no** se han vendido (LEFT JOIN).

select \* from productos p

left join ventas v

on p.id\_producto = v.id\_producto

where v.id\_venta is null;

1. Cuenta cuántos empleados hay en total.

select count(\*) as cantidad\_empleados from empleados;

1. Muestra el salario promedio de los empleados.

select avg(salario) as salario\_promedio from empleados;

1. Muestra el nombre del producto más caro.

select nombre, max(precio) as producto\_mas\_caro from productos;

1. Muestra el producto con el precio más bajo.

select nombre, min(precio) as producto\_mas\_barato from productos;

1. Calcula el total de ventas (suma de todos los precios × cantidades).

select p.nombre, (p.precio\*v.cantidad) as total\_de\_ventas from ventas v

join productos p

on v.id\_producto = p.id\_producto;

**Nivel Avanzado**

1. Crea una vista que muestre nombre\_cliente, nombre\_producto y fecha\_venta.

create view datos\_venta as

select c.nombre as nombre\_cliente, p.nombre as nombre\_producto, fecha\_venta from ventas v

join clientes c

on v.id\_cliente = c.id\_cliente

join productos p

on v.id\_producto = p.id\_producto;

1. Usa CASE para mostrar un campo "categoría precio" en productos:
   * Menor a 100 → "Barato"
   * 100 a 500 → "Moderado"
   * Mayor a 500 → "Caro"

select nombre,

case

when precio < 100 then 'barato'

when precio > 100 and precio < 500 then 'Moderado'

when precio > 500 then 'Caro'

else 'Sin Catalogar'

end as categoria\_precio

from productos;

1. Agrupa las ventas por mes (GROUP BY) y muestra el total vendido.

select p.nombre,date\_format(fecha\_venta, '%Y-%m')as mes,(p.precio\*v.cantidad) as total\_de\_ventas from ventas v

join productos p

on v.id\_producto = p.id\_producto

group by mes;

1. Muestra clientes que han comprado más de 2 veces (HAVING COUNT(\*) > 2).

select c.nombre, v.fecha\_venta, v.id\_producto from ventas v

join clientes c

on v.id\_cliente = c.id\_cliente

group by v.id\_cliente

having count(v.id\_cliente) > 2

1. Usa DATEDIFF para calcular cuánto tiempo lleva un empleado en la empresa.

select nombre, fecha\_ingreso, datediff(curdate(), fecha\_ingreso) as dias\_en\_la\_empresa from empleados;

1. Crea una subconsulta para mostrar el nombre del cliente que más ha comprado.
2. Crea una subconsulta que muestre el nombre del producto más vendido.
3. Crea una consulta que concatene el nombre y el país de cada cliente.

select concat(nombre, ' ', pais) as nombre\_pais from clientes;

1. Muestra los clientes que no han hecho ninguna compra (NOT EXISTS).
2. Muestra todos los productos ordenados de manera descendente por precio.

select nombre, precio from productos

order by precio desc;

**Nivel Experto**

1. Usa ROW\_NUMBER() OVER (ORDER BY salario DESC) para numerar los empleados por salario.

SELECT

nombre, salario,

ROW\_NUMBER() OVER (ORDER BY salario desc) AS numero\_fila

FROM

empleados;

1. Usa RANK() OVER (PARTITION BY departamento ORDER BY salario DESC) para mostrar el ranking de empleados por departamento.

SELECT

nombre, departamento,

rank() OVER (partition BY departamento order by salario desc) AS numero\_ranking

FROM

empleados;

1. Crea un procedimiento almacenado que inserte un nuevo cliente.

DELIMITER $$

CREATE PROCEDURE sp\_agregar\_cliente(

in p\_nombre varchar(100),

in p\_correo varchar(100),

in p\_pais varchar(50)

)

BEGIN

insert into Clientes (nombre,correo,pais)

values (p\_nombre,p\_correo,p\_pais);

END $$

DELIMITER ;

1. Crea un procedimiento almacenado que elimine un cliente por id.

DELIMITER $$

create procedure sp\_eliminar\_cliente

(in p\_id\_cliente int (11))

BEGIN

delete from clientes where id\_cliente = p\_id\_cliente;

END $$

DELIMITER ;

1. Crea un trigger que al insertar un nuevo producto, automáticamente lo deje con stock 0 si no se especifica.

DELIMITER $$

create trigger tr\_stock\_por\_defecto

before insert

on productos

for each row

BEGIN

if NEW.stock is null then

set NEW.stock = 0;

end if;

END $$

DELIMITER ;

1. Muestra las ventas realizadas en los últimos 30 días.

select id\_venta from ventas

where fecha\_venta = month(curdate());

1. Muestra el salario máximo, mínimo y promedio por departamento.

select departamento, max(salario) as salario\_maximo,min(salario)as salario\_minimo,avg(salario)as salario\_promedio

from empleados

group by departamento;

1. Muestra los 3 productos más vendidos (TOP 3).

select p.nombre, (v.cantidad\*p.precio) as ventas\_totales from productos p

join ventas v

on p.id\_producto = v.id\_producto

order by ventas\_totales desc

limit 3;

1. Muestra los 5 clientes que más han gastado dinero.

select c.id\_cliente,c.nombre,sum(v.cantidad\*p.precio) as total\_gastado from ventas v

join clientes c

on v.id\_cliente = c.id\_cliente

join productos p

on v.id\_producto = p.id\_producto

group by c.id\_cliente

order by total\_gastado desc

limit 5;

1. Haz una consulta que use COALESCE para reemplazar valores nulos en correo de cliente.

select nombre, coalesce(correo, 'Sin correo') as sin\_correo from clientes

where correo is null

**Reto Experto Final**

1. Crea una CTE (WITH) que liste los empleados y calcule su antigüedad.

WITH CTE\_calculo\_antiguedad AS (

SELECT

nombre,

fecha\_ingreso,

TIMESTAMPDIFF(YEAR, fecha\_ingreso, CURDATE()) AS años\_en\_la\_empresa

FROM empleados

)

SELECT \*

FROM CTE\_calculo\_antiguedad;

1. Usando CTE, muestra los productos cuyo precio es superior al promedio de todos los productos.

WITH CTE\_precio\_promedio AS (

SELECT AVG(precio) AS promedio

FROM productos

),

CTE\_productos\_con\_precio AS (

SELECT

p.nombre,

p.precio

FROM productos p

)

SELECT

c.nombre,

c.precio

FROM CTE\_productos\_con\_precio c, CTE\_precio\_promedio p

WHERE c.precio > p.promedio;

1. Usa una función de fecha para mostrar todos los empleados que ingresaron en año par.
2. Calcula cuántos días han pasado desde la última venta registrada.
3. Muestra, usando funciones de ventana, qué empleados ganan más que el promedio de su departamento.
4. Haz una subconsulta correlacionada para mostrar clientes que han comprado más de un tipo de producto.
5. Usa IN para mostrar todos los productos vendidos en enero o febrero.

SELECT p.nombre, v.fecha\_venta

FROM ventas v

JOIN productos p ON v.id\_producto = p.id\_producto

WHERE MONTH(v.fecha\_venta) IN (1, 2);

1. Usa LIKE para mostrar todos los clientes cuyo nombre contenga "an".

select \* from clientes

where nombre like '%an%';

1. Haz una consulta usando BETWEEN para encontrar ventas entre dos fechas específicas.

select v.id\_venta from ventas v

join productos p

on v.id\_producto = p.id\_producto

where v.fecha\_venta between '2020-06-10' and '2021-02-20';

1. Haz una consulta usando IS NULL para encontrar productos que no tienen stock registrado.

select \* from productos

where stock is null;