SERVICIO NACIONAL DE APRENDIZAJE SENA

Programa

ANÁLISIS Y DESARROLLO DE SOFTWARE

Ficha:

3069140

Título de La evidencia:

Joins, Subconsultas y Vistas

Instructor Responsable:

Javier Yara

Aprendiz:

Brenda Castiblanco

Camilo Hurtado

Juan Pablo Valderrama

Juan Camilo Gil

Robinson Almonacid

Mauren Gonzalez

SERVICIO NACIONAL DE APRENDIZAJE SENA

2025.

INNER JOIN..... 3
1.1 Clientes y Tipo de Documento..... 3
1.2 Productos y Categorías..... 4
1.3 Recibos y Clientes..... 4
1.4 Productos En recibo de caja con detalle..... 5
1.5 Usuarios y Roles..... 5
LEFT JOIN..... 6
2.1 Clientes y Usuario..... 6
2.3 Usuarios sin Roles..... 7
2.4 recibo de caja sin Cliente..... 7
2.5 Ventas con Productos Inexistentes..... 8
3.1 Usuarios sin Clientes..... 8
3.2 Categorías con o sin Productos..... 9
3.3 Roles sin Usuarios..... 9
3.4 Clientes sin recibo de caja..... 10
3.5 Productos sin Ventas..... 10
SUBCONSULTAS..... 11
4.1 Cliente con Más recibos de caja..... 11
4.2 Producto Más Vendido..... 12
4.3 Clientes con Alto Gasto..... 12
4.4 Productos Nunca Vendidos..... 13
4.5 Usuarios Sin Relación con Clientes..... 13
VISTAS..... 14
5.1 Vista de Clientes y Documentos..... 14
5.4 Vista de Usuarios y Roles..... 15
5.5 Vista de Detalle de Ventas..... 16
TRIGGERS..... 16
6.1 Reducir stock cuando se facture un producto..... 16
6.2 Validar stock antes de realizar recibo de caja..... 17
6.3 Registrar bitácora de cambios de usuarios..... 18
6.4 Calcular automáticamente precio de venta en productos_vendidos..... 19
6.5 Verificación de longitud mínima de contraseñas en usuarios..... 20
PROCEDIMIENTOS..... 21
7.1 Obtener recibos de caja de un cliente Devuelve todas los recibos de caja de un cliente específico..... 21
7.2 Procedimiento para actualizar el stock de un producto..... 22
7.3 Crear un nuevo cliente, Inserta un registro en la tabla clientes de forma controlada..... 22
7.4 Lista todos los productos activos/inactivos de una categoría específica..... 23
7.5 Generar detalle de recibos de caja -- Devuelve información completa de un recibo de caja: -- datos del cliente, productos registrados, precios y subtotales..... 24

FUNCIONES..... 25
8.1 Obtener nombre completo del cliente..... 25
8.2 Calcular total facturado de un cliente..... 26
8.3 Calcular stock valorizado de un producto..... 27
8.4 Obtener rol de un usuario.....28
8.5 Contar productos por categoría.....29
ENCRIPTACIÓN..... 30
9.1 encriptar contraseñas función.....30
9.2 Procedimiento para registrar usuario con contraseña encriptada.....31

INNER JOIN

1.1 **Clientes y Tipo de Documento.**

```
SELECT c.id_cliente, c.numero_documento, td.nombre_documento
FROM clientes c
INNER JOIN tipo_documento td ON c.id_tipo_documento = td.id_tipo_documento;
```

-- **Explicación**: Devuelve solo los clientes que tiene un tipo de documento válido registrado.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id_cliente	numero_documento	nombre_documento
1	1000123456	Cédula de Ciudadanía
4	1000123459	Cédula de Ciudadanía
7	1000123462	Cédula de Ciudadanía
10	1000123465	Cédula de Ciudadanía
13	1000123468	Cédula de Ciudadanía
16	1000123471	Cédula de Ciudadanía
19	1000123474	Cédula de Ciudadanía
2	1000123457	Cédula de Extranjería
5	1000123460	Cédula de Extranjería

Result 1

1.2 **Productos y Categorías**

```
SELECT p.id_producto, p.nombre_producto, c.nombre_categoria
FROM productos p
INNER JOIN categorias c ON p.id_categoria = c.id_categoria;
```

-- **Explicación**: Lista los productos con el nombre de su categoría.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id_producto	nombre_producto	nombre_categoria
1	Croquetas para perros adulto 5kg	Alimentos para perros
2	Alimento húmedo para gatos 400g	Alimentos para gatos
3	Snack dental para perros 150g	Alimentos para perros
4	Collar ajustable para perros	Accesorios para paseo
5	Juguete de cuerda para mascotas	Juguetes para perros
6	Cama ortopédica para perros medianos	Camas y descanso
7	Champú antipulgas para perros 500ml	Cuidado e higiene
8	Arena para gatos perfumada 10kg	Alimentos para gatos
9	Toallitas húmedas para mascotas 100 uds	Accesorios de limpieza

Result 2

1.3 **Recibos y Clientes**

```
SELECT
    r.id_recibo_caja,
    r.numero_recibo_caja,
    cl.primer_nombre,
    cl.primer_apellido
FROM
    recibo_caja r
INNER JOIN
    clientes cl ON r.id_cliente = cl.id_cliente;
```

-- **Explicación**: Muestra los recibos junto con el nombre del cliente que las generó.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id_recibo_caja	numero_recibo_caja	primer_nombre	primer_apellido
1	100001	Juan	Pérez
2	100002	María	López
3	100003	Carlos	Gómez
4	100004	Laura	Ramírez
5	100005	Andrés	Mendoza
6	100006	Juliana	Castillo
7	100007	Sebastián	Moreno
8	100008	Camila	Sánchez
9	100009	David	Rojas
10	100010	Valentina	Nieto
11	100011	Felipe	Vera
12	100012	Natalia	Acosta
13	100013	Santiago	León
14	100014	Daniela	Castro
15	100015	Mateo	Murcia
16	100016	Isabella	Soto
17	100017	Tomás	Ospina
18	100018	Sofía	Peña
19	100019	Lucas	Celis
20	100020	Martina	Guerrero

1.4 **Productos En recibo de caja con detalle**

SELECT id_producto_vendido, f.numero_recibo_caja, p.nombre_producto, pf.cantidad, pf.precio_venta

FROM productos_vendidos pf

INNER JOIN recibo_caja f ON pf.id_recibo_caja= f.id_recibo_caja

INNER JOIN productos p ON pf.id_producto = p.id_producto;

-- **Explicación**: Muestra qué productos fueron vendidos en cada recibo de caja..

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id_producto_vendido	numero_recibo_caja	nombre_producto	cantidad	precio_venta
1	100001	Croquetas para perros adulto 5kg	2	15000.00
2	100002	Alimento húmedo para gatos 400g	1	23999.99
3	100003	Snack dental para perros 150g	5	10240.00
4	100004	Collar ajustable para perros	3	26000.25
5	100005	Juguete de cuerda para mascotas	4	31250.00
6	100006	Cama ortopédica para perros medianos	2	22500.45
7	100007	Champú antipulgas para perros 500ml	1	67000.00
8	100008	Arena para gatos perfumada 10kg	3	9933.66
9	100009	Toallitas húmedas para mascotas 100 uds	2	45500.00
10	100010	Comida para peces perros 200g	1	74500.25
11	100011	Croquetas para perros adulto 5kg	2	25000.00
12	100012	Alimento húmedo para gatos 400g	4	22000.00
13	100013	Snack dental para perros 150g	3	10000.00
14	100014	Collar ajustable para perros	1	61200.50
15	100015	Juguete de cuerda para mascotas	2	24600.00
16	100016	Cama ortopédica para perros medianos	1	115000.00
17	100017	Champú antipulgas para perros 500ml	2	19450.00
18	100018	Arena para gatos perfumada 10kg	3	33333.33
19	100019	Toallitas húmedas para mascotas 100 uds	1	80500.75
20	100020	Comida para peces perros 200g	2	67000.00

1.5 **Usuarios y Roles**

SELECT u.id_usuario, u.correo_electronico, r.nombre_rol

FROM usuarios u

INNER JOIN usuario_rol ur ON u.id_usuario = ur.id_usuario

INNER JOIN roles r ON ur.id_rol = r.id_rol;

-- **Explicación**: Une usuarios con sus roles asignados.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id_usuario	correo_electronico	nombre_rol
1	admin@empresa.com	Administrador
3	user2@empresa.com	Administrador
5	user4@empresa.com	Administrador
7	user6@empresa.com	Administrador
9	user8@empresa.com	Administrador
11	user10@empresa.com	Administrador
13	user12@empresa.com	Administrador
15	user14@empresa.com	Administrador
17	user16@empresa.com	Administrador
19	user18@empresa.com	Administrador
2	user1@empresa.com	Vendedor
4	user3@empresa.com	Vendedor
6	user5@empresa.com	Vendedor
8	user7@empresa.com	Vendedor
10	user9@empresa.com	Vendedor
12	user11@empresa.com	Vendedor
14	user13@empresa.com	Vendedor
16	user15@empresa.com	Vendedor
18	user17@empresa.com	Vendedor
20	user19@empresa.com	Vendedor

LEFT JOIN

2.1 **Clientes y Usuario**

```
SELECT c.id_cliente, c.primer_nombre, u.correo_electronico
FROM clientes c
LEFT JOIN usuarios u ON c.id_usuario = u.id_usuario;
```

-- **Explicación**: Devuelve todos los clientes, incluso los que no tienen usuario en el sistema.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id_cliente	primer_nombre	correo_electronico
1	Juan	admin@empresa.com
2	María	user1@empresa.com
3	Carlos	NULL
4	Laura	NULL
5	Andrés	user2@empresa.com
6	Juliana	NULL
7	Sebastián	NULL
8	Camila	user3@empresa.com
9	David	NULL

Result 6

2.2 **Categorías sin Productos**

SELECT c.nombre_categoria, p.nombre_producto

FROM categorias c

LEFT JOIN productos p ON c.id_categoria = p.id_categoria;

-- **Explicación**: Devuelve todas las categorías, aunque no tengan productos.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

nombre_categoria	nombre_producto
Accesorios de limpieza	Toallitas húmedas para mascotas 100 uds
Accesorios de limpieza	Rascador para gatos mediano
Accesorios de limpieza	Correa retráctil para perros 5m
Accesorios de limpieza	Comedero automático para perros 3L
Accesorios para paseo	Collar ajustable para perros
Alimentos para gatos	Alimento húmedo para gatos 400g
Alimentos para gatos	Arena para gatos perfumada 10kg
Alimentos para gatos	Comida para gatos 1kg
Alimentos para gatos	Alimento balanceado para gatos 500g

Result 7

2.3 **Usuarios sin Roles**

SELECT u.correo_electronico, r.nombre_rol

FROM usuarios u

LEFT JOIN usuario_rol ur ON u.id_usuario = ur.id_usuario

LEFT JOIN roles r ON ur.id_rol = r.id_rol;

-- **Explicación**: Muestra todos los usuarios, aunque no tengan roles asignados.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

correo_electronico	nombre_rol
admin@empresa.com	Administrador
user10@empresa.com	Administrador
user11@empresa.com	Vendedor
user12@empresa.com	Administrador
user13@empresa.com	Vendedor
user14@empresa.com	Administrador
user15@empresa.com	Vendedor
user16@empresa.com	Administrador
user17@empresa.com	Vendedor

Result 8

2.4 **recibo de caja sin Cliente**

SELECT r.numero_recibo_caja, cl.primer_nombre, cl.primer_apellido

FROM recibo_caja r

LEFT JOIN clientes cl ON r.id_cliente = cl.id_cliente;

-- **Explicación**: Lista todas los recibos de caja, incluso si el cliente ya no existe.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

numero_recibo_caja	primer_nombre	primer_apellido
100001	Juan	Pérez
100002	María	López
100003	Carlos	Gómez
100004	Laura	Ramírez
100005	Andrés	Mendoza
100006	Juliana	Castillo
100007	Sebastián	Moreno
100008	Camila	Sánchez

2.5 **Ventas con Productos Inexistentes**

SELECT pf.id_producto_vendido, pf.cantidad, p.nombre_producto

FROM productos_vendidos pf

LEFT JOIN productos p ON pf.id_producto = p.id_producto;

-- **Explicación**: Devuelve todas las ventas registradas, aunque el producto ya no exista en inventario.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id_producto_vendido	cantidad	nombre_producto
1	2	Croquetas para perros adulto 5kg
2	1	Alimento húmedo para gatos 400g
3	5	Snack dental para perros 150g
4	3	Collar ajustable para perros
5	4	Juguete de cuerda para mascotas
6	2	Cama ortopédica para perros medianos
7	1	Champú antipulgas para perros 500ml
8	3	Arena para gatos perfumada 10kg

RIGHT JOIN

3.1 Usuarios sin Clientes

SELECT c.id_cliente, u.id_usuario, u.correo_electronico

FROM clientes c

RIGHT JOIN usuarios u ON c.id_usuario = u.id_usuario;

-- **Explicación**: Muestra todos los usuarios, incluso si no están asociados a un cliente.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id_cliente	id_usuario	correo_electronico
1	1	admin@empresa.com
NULL	11	user10@empresa.com
NULL	12	user11@empresa.com
NULL	13	user12@empresa.com
NULL	14	user13@empresa.com
NULL	15	user14@empresa.com
NULL	16	user15@empresa.com
NULL	17	user16@empresa.com
NULL	18	user17@empresa.com

Result 11

3.2 Categorías con o sin Productos

SELECT p.nombre_producto, c.nombre_categoria

FROM productos p

RIGHT JOIN categorias c ON p.id_categoria = c.id_categoria;

-- **Explicación**: Muestra todas las categorías, destacando incluso las que no tienen productos.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

nombre_producto	nombre_categoria
Toallitas húmedas para mascotas 100 uds	Accesorios de limpieza
Rascador para gatos mediano	Accesorios de limpieza
Correa retráctil para perros 5m	Accesorios de limpieza
Comedero automático para perros 3L	Accesorios de limpieza
Collar ajustable para perros	Accesorios para paseo
Alimento húmedo para gatos 400g	Alimentos para gatos
Arena para gatos perfumada 10kg	Alimentos para gatos
Comida para gatos 1kg	Alimentos para gatos
Alimento balanceado para gatos 500g	Alimentos para gatos

Result 12

3.3 Roles sin Usuarios

SELECT u.correo_electronico, r.nombre_rol

FROM usuarios u

RIGHT JOIN usuario_rol ur ON u.id_usuario = ur.id_usuario

RIGHT JOIN roles r ON ur.id_rol = r.id_rol;

-- **Explicación**: Lista todos los roles, incluso si no tienen usuarios.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

correo_electronico	nombre_rol
admin@empresa.com	Administrador
user2@empresa.com	Administrador
user4@empresa.com	Administrador
user6@empresa.com	Administrador
user8@empresa.com	Administrador
user10@empresa.com	Administrador
user12@empresa.com	Administrador
user14@empresa.com	Administrador
user16@empresa.com	Administrador

Result 13

3.4 Clientes sin recibo de caja

SELECT r.numero_recibo_caja, cl.primer_nombre

FROM recibo_caja r

RIGHT JOIN clientes cl ON r.id_cliente = cl.id_cliente;

-- **Explicación**: Lista todos los clientes, incluso si nunca han hecho un recibo de caja.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

numero_recibo_caja	primer_nombre
100001	Juan
100002	María
100003	Carlos
100004	Laura
100005	Andrés
100006	Juliana
100007	Sebastián
100008	Camila

Result 27

3.5 **Productos sin Ventas**

SELECT pf.id_producto_vendido, p.nombre_producto

FROM productos_vendidos pf

RIGHT JOIN productos p ON pf.id_producto = p.id_producto;

-- **Explicación**: Devuelve todos los productos, incluso si no se han vendido nunca.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id_producto_vendido	nombre_producto
1	Croquetas para perros adulto 5kg
11	Croquetas para perros adulto 5kg
2	Alimento húmedo para gatos 400g
12	Alimento húmedo para gatos 400g
3	Snack dental para perros 150g
13	Snack dental para perros 150g
4	Collar ajustable para perros
14	Collar ajustable para perros

Result 28

SUBCONSULTAS

4.1 Cliente con Más recibos de caja

```
SELECT id_cliente, COUNT(*) AS total_recibo_caja
FROM recibo_caja
GROUP BY id_cliente
HAVING COUNT(*) = (
   SELECT MAX(cantidad)
   FROM (SELECT COUNT(*) AS cantidad FROM recibo_caja GROUP BY id_cliente) AS t
);
-- Explicación: Encuentra al cliente que más recibos de caja tiene.
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id_cliente	total_recibo_caja
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	1

Result 29

4.2 Producto Más Vendido

```
SELECT id_producto, SUM(cantidad) AS total_vendido
FROM productos_vendidos
GROUP BY id_producto
HAVING SUM(cantidad) = (
   SELECT MAX(total)
   FROM (SELECT SUM(cantidad) AS total FROM productos_vendidos GROUP BY id_producto) t
);
-- Explicación: Devuelve el producto más vendido en la historia.
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id_producto	total_vendido
3	8

4.3 Clientes con Alto Gasto

SELECT id_cliente, SUM(total) AS gasto

FROM recibo_caja

GROUP BY id_cliente

HAVING SUM(total) > 100000;

-- **Explicación**: Subconsulta implícita dentro de agregación para detectar clientes grandes compradores.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id_cliente	gasto
1	150000.50
5	125000.00
16	115000.00
20	134000.00

4.4 Productos Nunca Vendidos

SELECT nombre_producto

FROM productos

WHERE id_producto NOT IN (SELECT DISTINCT id_producto FROM productos_vendidos);

-- **Explicación**: Lista los productos que nunca se han vendido.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

nombre_producto
Transportadora para gatos pequeña
Comida para gatos 1kg
Desodorante para mascotas spray 250ml
Rascador para gatos mediano
Leche para cachorros 1L
Cepillo para eliminación de pelo
Correa retráctil para perros 5m
Alimento balanceado para gatos 500g
Limpiador de oídos para mascotas 100ml

productos 20 ×

4.5 Usuarios Sin Relación con Clientes

SELECT correo_electronico

FROM usuarios

WHERE id_usuario NOT IN (SELECT id_usuario FROM clientes WHERE id_usuario IS NOT NULL);

-- **Explicación**: Encuentra usuarios que no tienen relación con clientes.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

correo_electronico
user10@empresa.com
user11@empresa.com
user12@empresa.com
user13@empresa.com
user14@empresa.com
user15@empresa.com
user16@empresa.com
user17@empresa.com
user18@empresa.com

usuarios 21

VISTAS

5.1 Vista de Clientes y Documentos

CREATE VIEW vw_clientes_documento AS

SELECT c.id_cliente, c.numero_documento, td.nombre_documento

FROM clientes c

INNER JOIN tipo_documento td ON c.id_tipo_documento = td.id_tipo_documento;

-- **Explicación**: Vista rápida para obtener clientes con su tipo de documento.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id_cliente	numero_documento	nombre_documento
1	1000123456	Cédula de Ciudadanía
4	1000123459	Cédula de Ciudadanía
7	1000123462	Cédula de Ciudadanía
10	1000123465	Cédula de Ciudadanía
13	1000123468	Cédula de Ciudadanía
16	1000123471	Cédula de Ciudadanía
19	1000123474	Cédula de Ciudadanía
2	1000123457	Cédula de Extranjería
5	1000123460	Cédula de Extranjería

_documento 1

5.2 **Vista de Productos y Categorías**

CREATE VIEW vw_productos_categoria AS

SELECT p.id_producto, p.nombre_producto, c.nombre_categoria

FROM productos p

INNER JOIN categorias c ON p.id_categoria = c.id_categoria;

-- **Explicación**: Muestra productos junto con su categoría.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id_producto	nombre_producto	nombre_categoria
1	Croquetas para perros adulto 5kg	Alimentos para perros
2	Alimento húmedo para gatos 400g	Alimentos para gatos
3	Snack dental para perros 150g	Alimentos para perros
4	Collar ajustable para perros	Accesorios para paseo
5	Juguete de cuerda para mascotas	Juguetes para perros
6	Cama ortopédica para perros medianos	Camas y descanso
7	Champú antipulgas para perros 500ml	Cuidado e higiene
8	Arena para gatos perfumada 10kg	Alimentos para gatos
9	Toallitas húmedas para mascotas 100 uds	Accesorios de limpieza

os_categoria 1

5.3 **Vista de Recibos de caja y Clientes**

CREATE VIEW vw_recibo_caja_clientes AS

SELECT f.numero_recibo_caja, f.total, cl.primer_nombre, cl.primer_apellido

FROM recibo_caja f

INNER JOIN clientes cl ON f.id_cliente = cl.id_cliente;

-- **Explicación**: Une recibo_caja con sus clientes.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

numero_recibo_caja	total	primer_nombre	primer_apellido
100001	150000.50	Juan	Pérez
100002	23999.99	María	López
100003	51200.00	Carlos	Gómez
100004	78000.75	Laura	Ramírez
100005	125000.00	Andrés	Mendoza
100006	45000.90	Juliana	Castillo
100007	67000.00	Sebastián	Moreno
100008	29800.99	Camila	Sánchez

Result 36

5.4 Vista de Usuarios y Roles

```
CREATE VIEW vw_usuarios_roles AS
SELECT u.correo_electronico, r.nombre_rol
FROM usuarios u
INNER JOIN usuario_rol ur ON u.id_usuario = ur.id_usuario
INNER JOIN roles r ON ur.id_rol = r.id_rol;
```

-- **Explicación**: Relación clara de usuarios y sus roles.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

correo_electronico	nombre_rol
admin@empresa.com	Administrador
user2@empresa.com	Administrador
user4@empresa.com	Administrador
user6@empresa.com	Administrador
user8@empresa.com	Administrador
user10@empresa.com	Administrador
user12@empresa.com	Administrador
user14@empresa.com	Administrador
user16@empresa.com	Administrador

uarios_roles 1

5.5 Vista de Detalle de Ventas

```
CREATE VIEW vw_detalle_ventas AS
SELECT f.numero_recibo_caja, p.nombre_producto, pf.cantidad, pf.precio_venta
FROM productos_vendidos pf
INNER JOIN recibo_caja  f ON pf.id_ = f.id_recibo_caja
INNER JOIN productos p ON pf.id_producto = p.id_producto;
```

-- **Explicación**: Vista que muestra el detalle de cada venta.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

numero_recibo_caja	nombre_producto	cantidad	precio_venta
100001	Croquetas para perros adulto 5kg	2	15000.00
100002	Alimento húmedo para gatos 400g	1	23999.99
100003	Snack dental para perros 150g	5	10240.00
100004	Collar ajustable para perros	3	26000.25
100005	Juguete de cuerda para mascotas	4	31250.00
100006	Cama ortopédica para perros medianos	2	22500.45
100007	Champú antipulgas para perros 500ml	1	67000.00
100008	Arena para gatos perfumada 10kg	3	9933.66

Result 39

TRIGGERS

6.1 **Reducir stock cuando se facture un producto**

```
DELIMITER //

CREATE TRIGGER tr_reducir_stock
AFTER INSERT ON productos_vendidos
FOR EACH ROW
BEGIN
    UPDATE productos
    SET stock = stock - NEW.cantidad
    WHERE id_producto = NEW.id_producto;
END //
DELIMITER //
```

```
265 •  CREATE TRIGGER tr_reducir_stock
266    AFTER INSERT ON productos_vendidos
267    FOR EACH ROW
268    BEGIN
269       UPDATE productos
270       SET stock = stock - NEW.cantidad
271       WHERE id_producto = NEW.id_producto;
272    END;
273    //
274
275    DELIMITER ;
276
277 •  SELECT id_producto, nombre_producto, stock
278    FROM productos;
279
280 •  INSERT INTO productos_vendidos (id_recibo_caja, id_producto, cantidad, precio_venta)
281    VALUES (2, 1, 3, 35990.00);
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

id_producto	nombre_producto	stock
1	Croquetas para perros adulto 5kg	47
2	Alimento húmedo para gatos 400g	100
3	Snack dental para perros 150g	75
4	Collar ajustable para perros	40
5	Juguete de cuerda para mascotas	60
6	Cama ortopédica para perros medianos	20
7	Champú antipulgas para perros 500ml	80
8	Arena para gatos perfumada 10kg	30
9	Toallitas húmedas para mascotas 100 uds	100
10	Comida para peces perros 200g	60
11	Transportadora para gatos pequeña	15
12	Comida para gatos 1kg	40

productos 2 ×

6.2 Validar stock antes de realizar recibo de caja

```
DELIMITER //

CREATE TRIGGER tr_validar_stock
BEFORE INSERT ON productos_vendidos
FOR EACH ROW
BEGIN
   DECLARE cantidad_disponible INT;

   -- Consultar el stock disponible del producto
   SELECT stock INTO cantidad_disponible
```

```
    FROM productos
    WHERE id_producto = NEW.id_producto;

    -- Validar si alcanza el stock
    IF cantidad_disponible < NEW.cantidad THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = '❌ No hay suficiente stock para este producto';
    END IF;
END //

DELIMITER ;
```

```
DELIMITER //

CREATE TRIGGER tr_validar_stock
BEFORE INSERT ON productos_vendidos
FOR EACH ROW
BEGIN
    DECLARE cantidad_disponible INT;

    -- Consultar el stock disponible del producto
    SELECT stock INTO cantidad_disponible
    FROM productos
    WHERE id_producto = NEW.id_producto;

    -- Validar si alcanza el stock
    IF cantidad_disponible < NEW.cantidad THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = '❌ No hay suficiente stock para este producto';
    END IF;
END //

DELIMITER ;
```

❌ No hay suficiente stock para este producto

6.3 Registrar bitácora de cambios de usuarios

```
DELIMITER //
CREATE TABLE IF NOT EXISTS bitacora_usuarios (
   id_log INT AUTO_INCREMENT PRIMARY KEY,
   id_usuario INT,
   accion VARCHAR(50),
   fecha TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE TRIGGER tr_log_update_usuario
AFTER UPDATE ON usuarios
FOR EACH ROW
BEGIN
   INSERT INTO bitacora_usuarios (id_usuario, accion)
   VALUES (NEW.id_usuario, 'Usuario actualizado');
END //
DELIMITER //
```

```
311 •   SELECT * FROM bitacora_usuarios;
312
313 •   UPDATE usuarios
314     SET idioma = 'Español'
315     WHERE id usuario = 1;
```

Result Grid | Filter Rows: | Edit:

id_log	id_usuario	accion	fecha
1	1	Usuario actualizado	2025-09-12 18:24:19
NULL	NULL	NULL	NULL

6.4 Calcular automáticamente precio de venta en productos vendidos

```
DELIMITER //
CREATE TRIGGER tr_set_precio_venta
BEFORE INSERT ON productos_vendidos
FOR EACH ROW
BEGIN
  IF NEW.precio_venta IS NULL OR NEW.precio_venta = 0 THEN
    SET NEW.precio_venta = (SELECT precio_unitario
                            FROM productos
                            WHERE id_producto = NEW.id_producto);
  END IF;
END //
DELIMITER //
```

```
316 •  CREATE TRIGGER trg_set_precio_venta
317    BEFORE INSERT ON productos_vendidos
318    FOR EACH ROW
319    BEGIN
320        DECLARE v_precio DECIMAL(10,2);
321
322        -- Buscar el precio unitario del producto
323        SELECT precio_unitario
324        INTO v_precio
325        FROM productos
326        WHERE id_producto = NEW.id_producto;
327
328        -- Asignar el precio de venta automáticamente
329        SET NEW.precio_venta = v_precio;
330    END$$
331
332    DELIMITER ;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

numero_recibo_caja	nombre_producto	cantidad	precio_venta	subtotal
100021	Croquetas para perros adulto 5kg	2	35.99	71.98
100021	Collar ajustable para perros	1	15.00	15.00

6.5 **Verificación de longitud mínima de contraseñas en usuarios**

```
DELIMITER //

CREATE TRIGGER tr_validar_password_length
BEFORE INSERT ON usuarios
FOR EACH ROW
BEGIN
   IF CHAR_LENGTH(NEW.contrasena) < 8 THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'La contraseña debe tener al menos 8 caracteres';
   END IF;
END//

DELIMITER ;
```

PROCEDIMIENTOS

7.1 **Obtener recibos de caja de un cliente**

Devuelve todas los recibos de caja de un cliente específico

```
DELIMITER //
CREATE PROCEDURE sp_recibos_cliente(IN p_id_cliente INT)
BEGIN
  SELECT
    rc.id_recibo_caja,
    rc.numero_recibo_caja,
    rc.fecha_recibo_caja,
    rc.total,
    rc.tipo_pago,
    c.primer_nombre,
    c.primer_apellido
  FROM recibo_caja rc
  INNER JOIN clientes c ON rc.id_cliente = c.id_cliente
  WHERE rc.id_cliente = p_id_cliente
  ORDER BY rc.fecha_recibo_caja DESC;
END//
DELIMITER ;
```

```
357  BEGIN
358      SELECT
359          rc.id_recibo_caja,
360          rc.numero_recibo_caja,
361          rc.fecha_recibo_caja,
362          rc.total,
363          rc.tipo_pago,
364          c.primer_nombre,
365          c.primer_apellido
366      FROM recibo_caja rc
367      INNER JOIN clientes c ON rc.id_cliente = c.id_cliente
368      WHERE rc.id_cliente = p_id_cliente
369      ORDER BY rc.fecha_recibo_caja DESC;
370  END//
371
372  DELIMITER ;
373
374  CALL sp_recibos_cliente(1);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id_recibo_caja	numero_recibo_caja	fecha_recibo_caja	total	tipo_pago	primer_nombre	primer_apellido
21	100021	2025-10-02 17:11:16	0.00	efectivo	Juan	Pérez
1	100001	2025-10-02 12:35:58	150000.50	efectivo	Juan	Pérez

7.2 **Procedimiento para actualizar el stock de un producto**

```
DELIMITER //
CREATE PROCEDURE sp_actualizar_stock(IN p_id_producto INT, IN p_cantidad INT)
BEGIN
   UPDATE productos
   SET stock = stock + p_cantidad
   WHERE id_producto = p_id_producto;
END //
DELIMITER ;
```

```
442     DELIMITER //
443     CREATE PROCEDURE sp_actualizar_stock(IN p_id_producto INT, IN p_cantidad INT)
444     BEGIN
445         UPDATE productos
446         SET stock = stock + p_cantidad
447         WHERE id_producto = p_id_producto;
448     END //
449     DELIMITER ;
450     CALL sp_actualizar_stock(1, 10);
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Con

id_producto	stock
1	70
NULL	NULL

7.3 **Crear un nuevo cliente, Inserta un registro en la tabla clientes de forma controlada**

```
DELIMITER //
CREATE PROCEDURE sp_crear_cliente(
   IN p_numero_documento VARCHAR(20),
   IN p_correo VARCHAR(150),
   IN p_primer_nombre VARCHAR(100),
```

```
    IN p_segundo_nombre VARCHAR(100),
    IN p_primer_apellido VARCHAR(100),
    IN p_segundo_apellido VARCHAR(100),
    IN p_telefono VARCHAR(20),
    IN p_id_tipo_documento INT,
    IN p_id_usuario INT
)
BEGIN
    INSERT INTO clientes (
        numero_documento, correo_electronico, primer_nombre, segundo_nombre,
        primer_apellido, segundo_apellido, numero_telefono, id_tipo_documento,
id_usuario
    )
    VALUES (
        p_numero_documento, p_correo, p_primer_nombre, p_segundo_nombre,
        p_primer_apellido, p_segundo_apellido, p_telefono, p_id_tipo_documento,
p_id_usuario
    );
END //
DELIMITER ;
```

```
490 • CALL sp_crear_cliente(
491       '1234567890', 'nuevo.cliente@correo.com', 'Pedro', 'Luis',
492       'Ramírez', 'Torres', '3101234567', 1, NULL
493   );
494 • SELECT * FROM clientes ORDER BY id cliente DESC LIMIT 1;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Fetch rows:

id_cliente	numero_documento	correo_electronico	primer_nombre	segundo_nombre	primer_apellido	segundo_apellido	numero_telefono	id_usuario	id_tipo_documento	fecha_creacion	fecha_actualizacion
21	1234567890	nuevo.cliente@correo.com	Pedro	Luis	Ramírez	Torres	3101234567	NULL	1	2025-09-12 18:40:50	2025-09-12 18:40:50
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

7.4 Lista todos los productos activos/inactivos de una categoría específica

```
DELIMITER //
CREATE PROCEDURE sp_productos_por_categoria(IN p_id_categoria INT)
BEGIN
   SELECT p.id_producto, p.nombre_producto, p.precio_unitario, p.stock, p.estado
   FROM productos p
   WHERE p.id_categoria = p_id_categoria;
END //
DELIMITER ;
```

```
497  DELIMITER //
498  CREATE PROCEDURE sp_productos_por_categoria(IN p_id_categoria INT)
499  BEGIN
500      SELECT p.id_producto, p.nombre_producto, p.precio_unitario, p.stock, p.estado
501      FROM productos p
502      WHERE p.id_categoria = p_id_categoria;
503  END //
504  DELIMITER ;
505  CALL sp_productos_por_categoria(1);
506
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id_producto	nombre_producto	precio_unitario	stock	estado
1	Croquetas para perros adulto 5kg	35.99	70	activo
3	Snack dental para perros 150g	8.99	75	activo
10	Comida para peces perros 200g	5.99	60	activo
15	Leche para cachorros 1L	20.00	30	activo

7.5 Generar detalle de recibos de caja -- Devuelve información completa de un recibo de caja: -- datos del cliente, productos registrados, precios y subtotales

```
DELIMITER //

CREATE PROCEDURE sp_recibo_detalle(IN p_id_recibo_caja INT)
BEGIN
  SELECT
    rc.numero_recibo_caja,
    rc.fecha_recibo_caja,
    CONCAT(c.primer_nombre, ' ', IFNULL(c.segundo_nombre,''), ' ',
        c.primer_apellido, ' ', IFNULL(c.segundo_apellido,'')) AS cliente,
    p.nombre_producto,
    pv.cantidad,
    pv.precio_venta,
    (pv.cantidad * pv.precio_venta) AS subtotal,
    rc.total AS total_recibo
  FROM recibo_caja rc
  INNER JOIN clientes c ON rc.id_cliente = c.id_cliente
  INNER JOIN productos_vendidos pv ON rc.id_recibo_caja = pv.id_recibo_caja
  INNER JOIN productos p ON pv.id_producto = p.id_producto
  WHERE rc.id_recibo_caja = p_id_recibo_caja;
END//

DELIMITER ;
```

```
383         rc.fecha_recibo_caja,
384         CONCAT(c.primer_nombre, ' ', IFNULL(c.segundo_nombre,''), ' ',
385               c.primer_apellido, ' ', IFNULL(c.segundo_apellido,'')) AS cliente,
386         p.nombre_producto,
387         pv.cantidad,
388         pv.precio_venta,
389         (pv.cantidad * pv.precio_venta) AS subtotal,
390         rc.total AS total_recibo
391     FROM recibo_caja rc
392     INNER JOIN clientes c ON rc.id_cliente = c.id_cliente
393     INNER JOIN productos_vendidos pv ON rc.id_recibo_caja = pv.id_recibo_caja
394     INNER JOIN productos p ON pv.id_producto = p.id_producto
395     WHERE rc.id_recibo_caja = p_id_recibo_caja;
396 END//
397
398 DELIMITER ;
399
400 CALL sp_recibo_detalle(1);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

numero_recibo_caja	fecha_recibo_caja	cliente	nombre_producto	cantidad	precio_venta	subtotal	total_recibo
100001	2025-10-02 12:35:58	Juan Pérez	Croquetas para perros adulto 5kg	53	15000.00	795000.00	150000.50

FUNCIONES

8.1 **Obtener nombre completo del cliente**

```
-- Devuelve primer_nombre + primer_apellido
DELIMITER //
CREATE FUNCTION fn_nombre_cliente(p_id_cliente INT)
RETURNS VARCHAR(255)
DETERMINISTIC
BEGIN
   DECLARE nombre_completo VARCHAR(255);

   SELECT CONCAT(primer_nombre, ' ', primer_apellido)
   INTO nombre_completo
   FROM clientes
   WHERE id_cliente = p_id_cliente;

   RETURN nombre_completo;
END //
```

DELIMITER ;

```
532     DELIMITER //
533 •   CREATE FUNCTION fn_nombre_cliente(p_id_cliente INT)
534     RETURNS VARCHAR(255)
535     DETERMINISTIC
536     BEGIN
537         DECLARE nombre_completo VARCHAR(255);
538
539         SELECT CONCAT(primer_nombre, ' ', primer_apellido)
540         INTO nombre_completo
541         FROM clientes
542         WHERE id_cliente = p_id_cliente;
543
544         RETURN nombre_completo;
545     END //
546     DELIMITER ;
547
548 •   SELECT fn_nombre_cliente(1) AS nombre_cliente;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content

nombre_cliente
Juan Pérez

8.2 **Calcular total comprado por un cliente**

-- Devuelve la suma de todos sus recibos de caja

DELIMITER //

CREATE FUNCTION fn_total_recaudado_cliente(p_id_cliente INT)

RETURNS DECIMAL(12,2)

DETERMINISTIC

BEGIN

DECLARE total_cliente DECIMAL(12,2);

SELECT COALESCE(SUM(total), 0)

INTO total_cliente

FROM recibo_caja

WHERE id_cliente = p_id_cliente;

RETURN total_cliente;

END//

DELIMITER ;

```
407 •  CREATE FUNCTION fn_total_recaudado_cliente(p_id_cliente INT)
408    RETURNS DECIMAL(12,2)
409    DETERMINISTIC
410    BEGIN
411        DECLARE total_cliente DECIMAL(12,2);
412
413        SELECT COALESCE(SUM(total), 0)
414        INTO total_cliente
415        FROM recibo_caja
416        WHERE id_cliente = p_id_cliente;
417
418        RETURN total_cliente;
419    END//
420
421    DELIMITER ;
422
423 •  SELECT fn_total_recaudado_cliente(1) AS total_cliente;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

total_cliente
150000.50

8.3 **Calcular stock valorizado de un producto**

-- Devuelve stock * precio_unitario

DELIMITER //

CREATE FUNCTION fn_stock_valorizado(p_id_producto INT)

RETURNS DECIMAL(12,2)

DETERMINISTIC

BEGIN

DECLARE valor DECIMAL(12,2);

SELECT stock * precio_unitario

INTO valor

FROM productos

```
    WHERE id_producto = p_id_producto;

    RETURN valor;
END //
DELIMITER ;
```

```
574     DELIMITER //
575     CREATE FUNCTION fn_stock_valorizado(p_id_producto INT)
576     RETURNS DECIMAL(12,2)
577     DETERMINISTIC
578     BEGIN
579         DECLARE valor DECIMAL(12,2);
580
581         SELECT stock * precio_unitario
582         INTO valor
583         FROM productos
584         WHERE id_producto = p_id_producto;
585
586         RETURN valor;
587     END //
588     DELIMITER ;
589     SELECT fn_stock_valorizado(1) AS stock_valorizado;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

stock_valorizado
2519.30

8.4 **Obtener rol de un usuario**

-- Devuelve el rol principal asociado a un usuario

```
DELIMITER //
CREATE FUNCTION fn_rol_usuario(p_id_usuario INT)
RETURNS VARCHAR(100)
DETERMINISTIC
BEGIN
    DECLARE rol_nombre VARCHAR(100);
```

```
    SELECT r.nombre_rol
    INTO rol_nombre
    FROM roles r
    INNER JOIN usuario_rol ur ON r.id_rol = ur.id_rol
    WHERE ur.id_usuario = p_id_usuario
    LIMIT 1;

    RETURN rol_nombre;
END //
DELIMITER ;
```

```
596     DELIMITER //
597     CREATE FUNCTION fn_rol_usuario(p_id_usuario INT)
598     RETURNS VARCHAR(100)
599     DETERMINISTIC
600     BEGIN
601         DECLARE rol_nombre VARCHAR(100);
602
603         SELECT r.nombre_rol
604         INTO rol_nombre
605         FROM roles r
606         INNER JOIN usuario_rol ur ON r.id_rol = ur.id_rol
607         WHERE ur.id_usuario = p_id_usuario
608         LIMIT 1;
609
610         RETURN rol_nombre;
611     END //
612     DELIMITER ;
613     SELECT fn_rol_usuario(1) AS rol_usuario;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Conten

rol_usuario
Administrador

8.5 **Contar productos por categoría**

```
-- Devuelve cuántos productos hay en una categoría
DELIMITER //
CREATE FUNCTION fn_total_productos_categoria(p_id_categoria INT)
RETURNS INT
DETERMINISTIC
```

```
BEGIN
    DECLARE total INT;

    SELECT COUNT(*)
    INTO total
    FROM productos
    WHERE id_categoria = p_id_categoria;

    RETURN total;
END //
DELIMITER ;
```

```
619   DELIMITER //
620   CREATE FUNCTION fn_total_productos_categoria(p_id_categoria INT)
621   RETURNS INT
622   DETERMINISTIC
623   BEGIN
624       DECLARE total INT;
625       SELECT COUNT(*)
626       INTO total
627       FROM productos
628       WHERE id_categoria = p_id_categoria;
629       RETURN total;
630   END //
631
632   use petshop;
633   DELIMITER ;
634   SELECT fn_total_productos_categoria(1) AS total_productos;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

total_productos
4

ENCRIPTACIÓN.

9.1 encriptar contraseñas función

```
DELIMITER //
CREATE FUNCTION fn_encriptar(p_pass VARCHAR(255))
RETURNS VARCHAR(255)
```

```
DETERMINISTIC
BEGIN
  RETURN SHA2(p_pass,256);
END //
DELIMITER ;
INSERT INTO usuarios (contrasena, correo_electronico)
VALUES (fn_encriptar('MiPassword123'), 'user@correo.com');
ALTER TABLE usuarios
MODIFY contrasena VARCHAR(256) NOT NULL;
#Cuando un usuario se cree, la contraseña se convertirá automáticamente en hash:
DELIMITER /
```

9.2 **Procedimiento para registrar usuario con contraseña encriptada**

```
CREATE PROCEDURE RegistrarUsuario(
  IN p_username VARCHAR(100),
  IN p_password VARCHAR(255),
  IN p_email VARCHAR(150),
  IN p_nombre VARCHAR(100) )
  BEGIN
  DECLARE v_salt VARCHAR(32);
  DECLARE v_encrypted VARCHAR(64);
  DECLARE v_user_exists INT DEFAULT 0;
  DECLARE i INT DEFAULT 0;

  -- Verificar si el usuario ya existe
  SELECT COUNT(*) INTO v_user_exists
  FROM usuarios
  WHERE username = p_username OR email = p_email;
  IF v_user_exists > 0 THEN
```

```
    SELECT "ERROR: USUARIO O EMAIL YA EXISTE" AS mensaje, FALSE AS exito;
  ELSE
   SET v_salt = SUBSTRING(MD5(CONCAT(RAND(), NOW())), 1, 32);
    -- Hash inicial con salt
    SET v_encrypted = SHA2(CONCAT(v_salt, p_password), 256);
    -- Simulación de múltiples rondas de hash (ajustable)
    SET i = 0;
    WHILE i < 10 DO
      SET v_encrypted = SHA2(CONCAT(v_salt, v_encrypted), 256);
      SET i = i + 1;
    END WHILE;
    -- Insertar usuario
    INSERT INTO usuarios (username, password_hash, salt, email, nombre)
    VALUES (p_username, v_encrypted, v_salt, p_email, p_nombre);
    SELECT "USUARIO REGISTRADO EXITOSAMENTE" AS mensaje, TRUE AS exito;
  END IF;
END //
DELIMITER ; select *
from usuarios;
```