

# FRISSE



PROYECTO "COMETA"

BASES DE DATOS

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO (UNAM)



# EQUIPO



DENISSE MARTÍNEZ

CEO Y DESARROLLADORA DE SOFTWARE

FRIDA MARTÍNEZ

ACCIONISTA Y DESARROLLADORA DE  
SOFTWARE

# MISIÓN

En FRISSE queremos ayudarte a maximizar el potencial de tu compañía en su proceso de Transformación Digital. Lo hacemos desarrollando e implementando soluciones punta a punta integradas por los mejores expertos y las mejores tecnologías especializados en creación de bases datos y páginas web.



# VISIÓN

66

"Proveer la mejor experiencia tecnológica al cliente cada día."

# DESCRIPCIÓN DE PROYECTO

## BASE DE DATOS

Desarrollo de sistemas informáticos para satisfacer requerimientos de una cadena de papelerías que busca innovar la manera en que almacena su información. Además de la creación de la interfaz gráfica para la interacción entre la Base de Datos y el usuario.



# REQUERIMIENTOS

## ALMACENAMIENTO

Almacenar información de productos, ventas, clientes y proveedores.

## GENERACIÓN DE FACTURAS

Creación automática de factura dado un número de venta.

## UTILIDAD

Dado un código de barra, obtener la utilidad de dicho producto.

## ÍNDICE

Creación de índice para agilizar consultas.

## STOCK

Obtener los productos que estén registrados con un stock menor a 3.

## VENTAS

Dada una fecha o fechas, obtener la cantidad vendida.

## DECREMENTO DE STOCK

Decremento de cada stock basado en la venta. Si el producto está por debajo del estandar, lanzar mensaje, o si el producto está agotado no realizar venta.

# SOFTWARE



## Trello

Software de administración de proyectos con interfaz web

## draw.io

Herramienta de creación y edición de diagramas libre.

## Overleaf

Editor colaborativo de LaTex que se utiliza para escribir, editar y publicar textos

## PostgreSQL

pgAdmin 3 es una herramienta de código abierto para la administración de bases de datos PostgreSQL y derivado.

## Visual Studio Code

Es un editor de código fuente desarrollado por Microsoft para Windows, Linux y macOS.

## XAMPP

Distribución de Apache completamente gratuita y fácil de instalar que contiene MariaDB, PHP y Perl

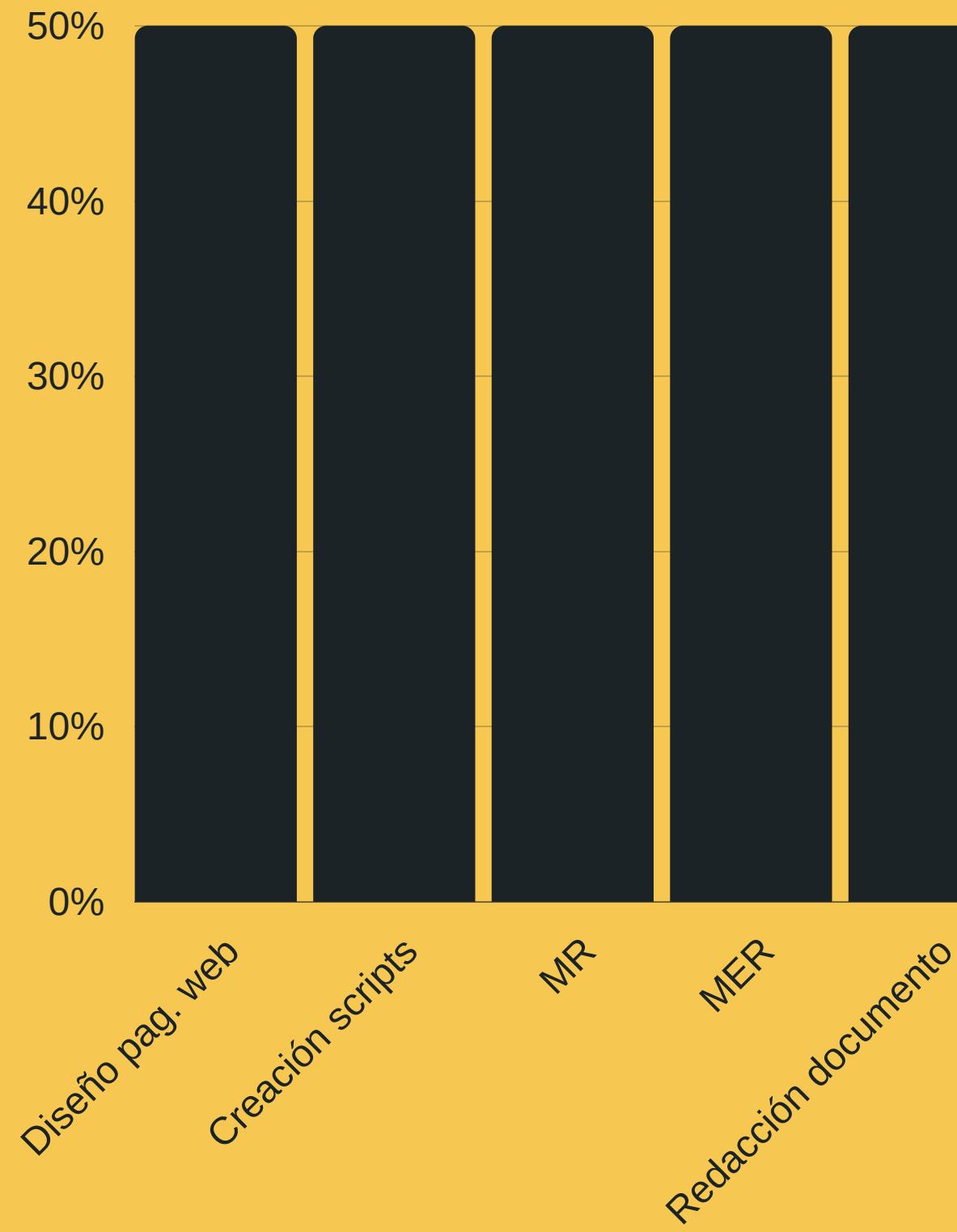
# PLAN DE TRABAJO

The image shows a Trello board titled "Bases de Datos". The board has five columns:

- Lista de Tareas - FASE 1**: Contains cards for "Link" (due 16 dic. de 2020) and "MODELO ENTIDAD RELACIÓN" (due 16 dic. de 2020).
- CÓDIGO - FASE 2**: Contains cards for "Creación de tablas" and "Llenado de tablas".
- PÁGINA WEB - FASE 3**: Contains cards for "Página Web" (due 19 de ene.) and "ENTREGABLE" (due 19 de ene.).
- LaTeX**: Contains a card for "ENTREGABLE" (due 19 de ene.).
- Presentación**: Contains a card for "Presentación POWER POINT" (due 19 de ene.).

Each card includes a progress bar, a due date, and a comment section.

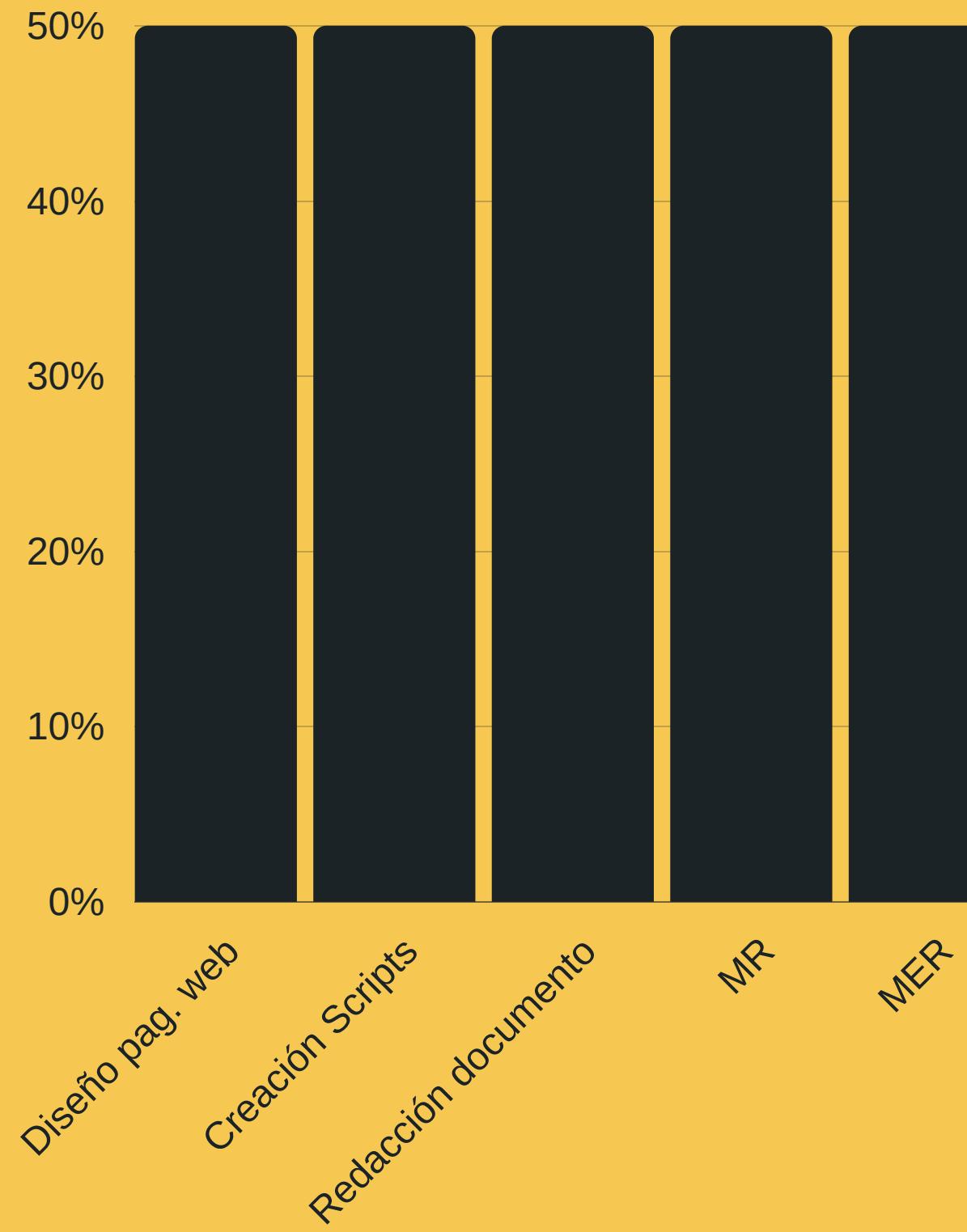
# ACTIVIDAD



## DENISSE MARTÍNEZ

Diseño de la página web, creación de scripts de la base de datos, programación de funciones como lo fue obtención de utilidad, obtención de número de ventas según fecha o fechas, diseño de MR, MER.

# ACTIVIDAD



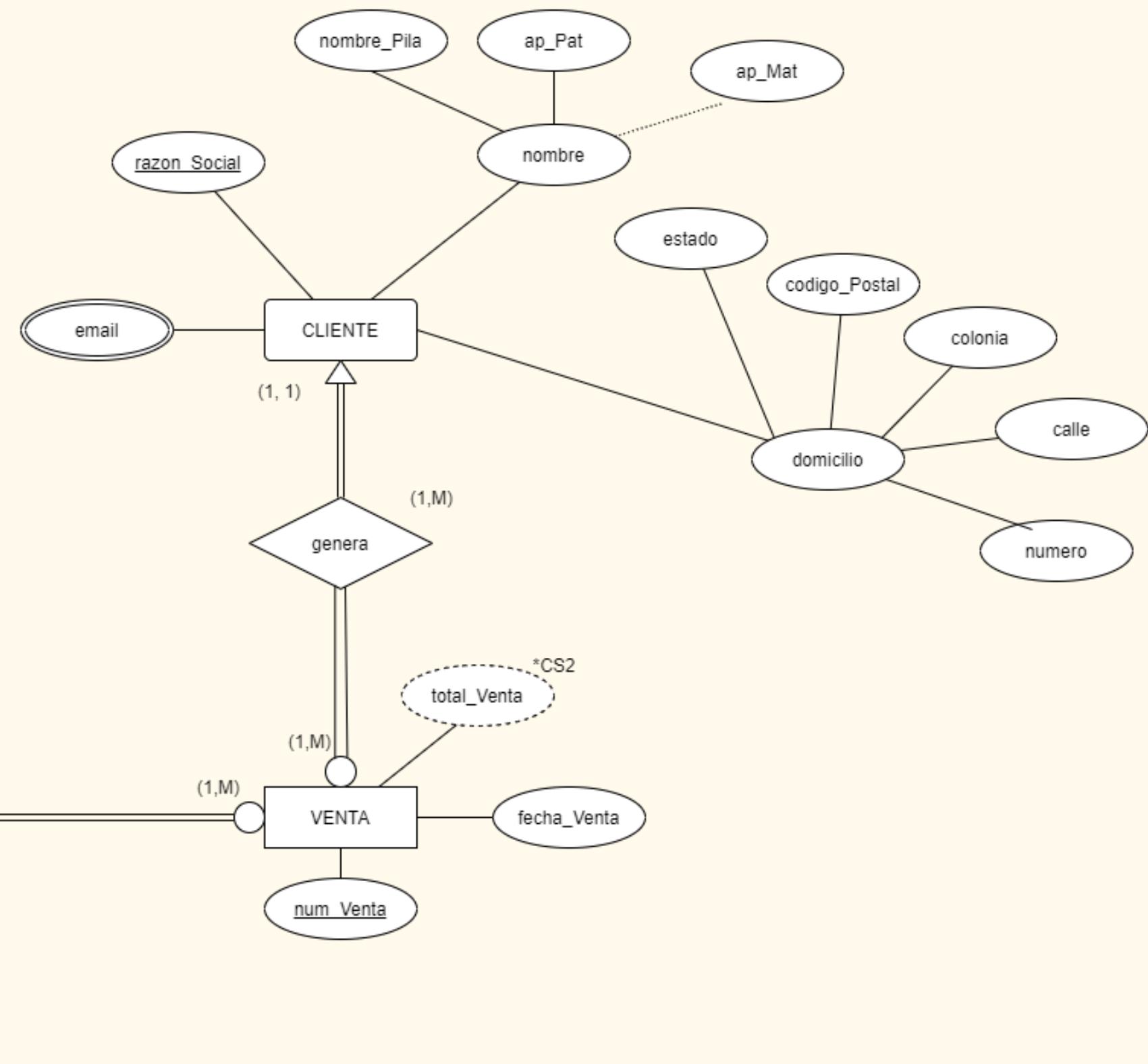
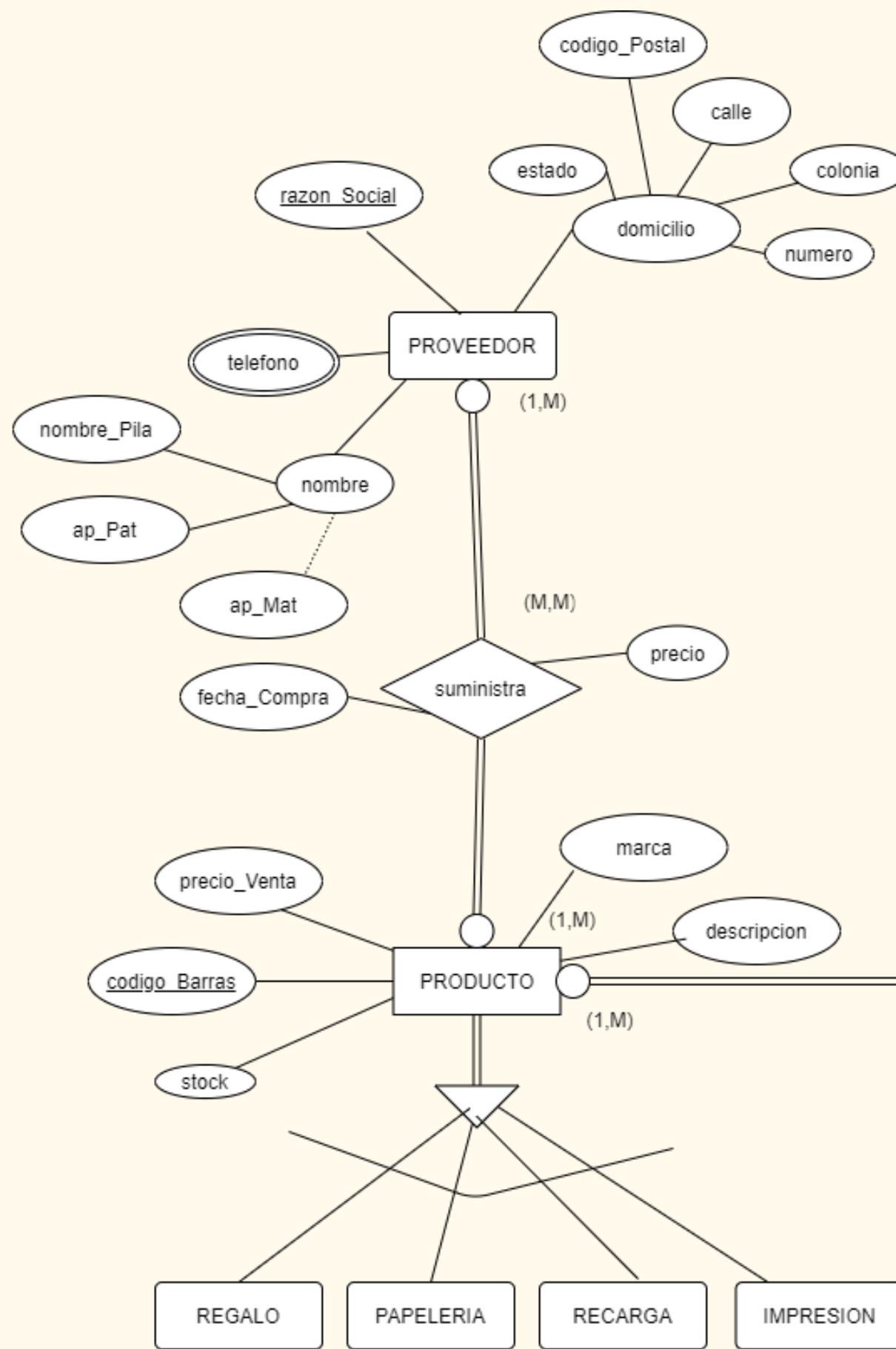
## FRIDA MARTÍNEZ

Diseño de MR y MER, creación de scripts de llenado de información y programación (stock, generación de vista de factura), diseño de pagina web, redacción del documento por escrito.

# DIS-<sup>~</sup>NO

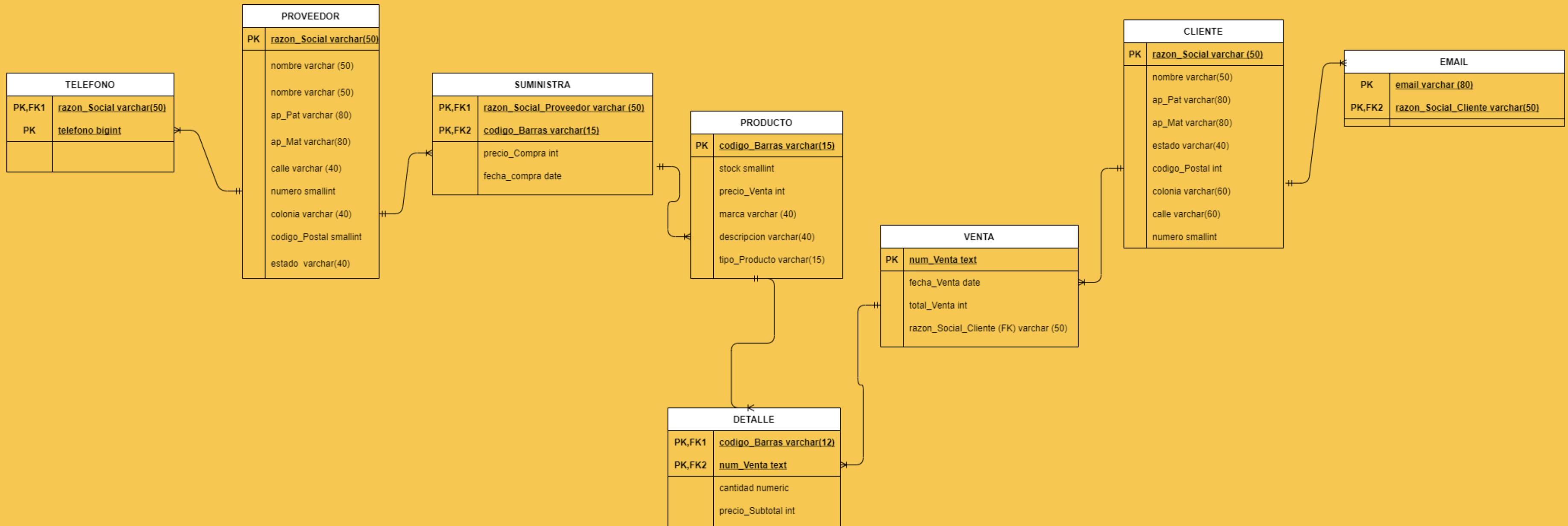


# MODELO ENTIDAD RELACIÓN



CS1 = Se calcula a partir de el precio y la cantidad.  
 CS2 = Se calcula a partir de la suma del los precios totales por articulo  
 CS3 = Menor o igual a 3

# MODELO RELACIONAL





# Base de Datos

CODIFICACIÓN

# TABLAS

```
CREATE TABLE VENTA(
    no_Venta TEXT NOT NULL DEFAULT 'VENT-0' || nextval('noVenta'::regclass)::TEXT,
    fecha_Venta date not null default current_date,
    total_Venta int,
    razon_Social varchar(50) not null,
    CONSTRAINT PK_Venta PRIMARY KEY (no_Venta),
    CONSTRAINT FK_Razon_Social_Client FOREIGN KEY (razon_Social) REFERENCES PUBLIC.CLIENTE (razon_Social)
);
```

```
create table PRODUCTO(
    codigo_Barras varchar(15) not null,
    stock smallint not null,
    precio_Venta int not null,
    marca varchar(40) not null,
    descripcion varchar(40) not null,
    tipo_Producto varchar(15) not null,
    CONSTRAINT PK_Producto PRIMARY KEY (codigo_Barras),
    CONSTRAINT verifica_Precio CHECK((CAST (precio_Venta AS numeric))> 0),
    CONSTRAINT tipo_Producto CHECK(tipo_Producto = 'REGALO' OR tipo_Producto= 'PAPELERIA' or tipo_Producto='RECARGA' or tipo_Producto='IMPRESION'),
    CONSTRAINT verifica_Stock CHECK (stock >= 0)
);
```

## TRIGGER SUBTOTAL

```
CREATE OR REPLACE FUNCTION calculaSubtotal() RETURNS TRIGGER AS $$  
DECLARE  
no_VentaAct text;  
barras varchar(15);  
subtotal int;  
total int;  
BEGIN  
  
no_VentaAct= 'VENT-0'||currval('noventa'::regclass)::text;  
  
barras = (select det.codigo_barras  
FROM PRODUCTO prod  
INNER JOIN DETALLE det ON (det.codigo_Barras = prod.codigo_Barras)  
where det.no_Venta = no_VentaAct and det.total_prod is null);  
  
subtotal= (select det.cantidad * prod.precio_Venta  
FROM PRODUCTO prod  
INNER JOIN DETALLE det ON (det.codigo_Barras = prod.codigo_Barras)  
where det.no_Venta = no_VentaAct and det.total_prod is null);  
  
IF(TG_OP= 'INSERT' )then  
UPDATE DETALLE SET total_Prod = subtotal where detalle.total_prod is null and no_Venta= no_VentaAct;  
  
total=(select SUM(total_Prod) FROM DETALLE det INNER JOIN VENTA vent ON(det.no_Venta = vent.no_venta)  
where vent.no_Venta= no_VentaAct);  
  
UPDATE VENTA SET total_Venta =total WHERE no_Venta = no_VentaAct;  
RETURN new;  
end if;  
  
END;  
$$ language plpgsql VOLATILE;
```

Esta función que de trigger hace un cálculo automático de los **subtotales** de cada una de las ventas. Además de calcular el **total** de la venta actual.

```
CREATE TRIGGER calcula_Subtotal  
AFTER INSERT ON DETALLE FOR EACH ROW  
EXECUTE PROCEDURE calculaSubtotal();
```

# GENERACIÓN DE FACTURAS

```
CREATE OR REPLACE FUNCTION factura(numVenta text) RETURNS void AS $$  
DECLARE contador int DEFAULT 0;  
DECLARE registro record;  
DECLARE registro1 record;  
DECLARE contador1 int DEFAULT 0;  
DECLARE registro2 record;  
  
BEGIN  
    FOR registro IN select fecha_Venta as FECHA, nextval('noFactura') as FACTURA, cli.razon_Social as compañía, concat(cli.nombre,' ',ap_Pat,' ', ap_Mat) as NOMBRE,  
        concat(calle, ' ',numero, ' ', colonia, ' ', estado, ' ', codigo_Postal ) AS DIRECCION  
    FROM VENTA vent  
    INNER JOIN CLIENTE cli ON(vent.razon_Social = cli.razon_Social)  
    WHERE no_Venta = numVenta LOOP  
        contador = contador+1;  
        RAISE NOTICE 'Fecha: %', registro.fecha;  
        RAISE NOTICE 'No Factura: %', registro.factura;  
        RAISE NOTICE 'Razon Social: %', registro.compañía;  
        RAISE NOTICE 'Nombre: %', registro.nombre;  
        RAISE NOTICE 'Direccion: %', registro.direccion;  
    END LOOP;  
    RAISE NOTICE '';  
    RAISE NOTICE 'DESCRIPCION      PRECIO UNITARIO      CANTIDAD      TOTAL PROD';  
    FOR registro1 IN select pro.descripcion AS descripcion, pro.precio_venta as precio_Unitario, det.cantidad as cantidad, det.total_Prod as totalProd FROM DETALLE det  
    INNER JOIN PRODUCTO pro ON (det.codigo_Barras = pro.codigo_Barras)  
    WHERE no_Venta = numVenta LOOP  
        RAISE NOTICE '%      %      %      %%', registro1.descripcion, registro1.precio_Unitario, registro1.cantidad, registro1.totalProd;  
        contador1=contador1+1;  
    END LOOP;  
  
    RAISE NOTICE '';  
  
    FOR registro2 IN SELECT total_Venta FROM VENTA WHERE VENTA.no_Venta = numVenta LOOP  
        RAISE NOTICE 'Total= $%', registro2.total_Venta;  
    END LOOP;  
END;  
$$  
LANGUAGE plpgsql;
```

# UTILIDAD

Esta función realiza la operación de obtener la utilidad de cada uno de los productos que están a la venta en la papelería.

```
CREATE OR REPLACE FUNCTION obten_utilidad(codigo_Bar varchar(15)) returns int
as
$$
    select prod.precio_Venta - sumi.precio_Compra FROM PRODUCTO prod
INNER JOIN SUMINISTRA sumi ON (prod.codigo_Barras = sumi.codigo_Barras)
where prod.codigo_Barras=codigo_Bar;
$$
language sql;
```

## ÍNDICE

Se crea un índice en la tabla de producto, esto con el fin de agilizar las consultas que se encuentran en esta tabla.

```
CREATE INDEX inx_stock ON PRODUCTO (stock);
```

## STOCK

Función que obtiene los productos con existencia menor a tres.

```
CREATE OR REPLACE FUNCTION obtenStock() RETURNS TABLE(producto varchar(20)) AS $$  
BEGIN  
    RETURN QUERY EXECUTE  
        format($f$SELECT descripcion FROM PRODUCTO WHERE stock<3$f$);  
END $$  
language plpgsql VOLATILE;
```

# VENTAS

Dada una fecha o un periodo de tiempo, obtener la cantidad vendida.

```
/*Dada una fecha regresar la cantidad total
que se vendio en esa fecha*/
CREATE OR REPLACE FUNCTION obten_CantidadVendida(fecha_Esp date) returns integer
as
$$
    select SUM(det.cantidad) FROM VENTA vent
    FULL JOIN DETALLE det ON (det.no_Venta = vent.no_Venta)
    where fecha_Venta = fecha_Esp GROUP BY fecha_Venta;
$$96+857+69
language sql;
--Llamada a la función
SELECT obten_CantidadVendida ('2020-07-28');

/*
Obtiene la cantidad vendida en un
periodo de tiempo*/

CREATE OR REPLACE FUNCTION obten_CantidadVendida(fecha1 date, fecha2 date) returns integer
AS
$$
    select SUM(det.cantidad) FROM VENTA vent
    INNER JOIN DETALLE det ON (det.no_Venta = vent.no_Venta)
    WHERE vent.fecha_Venta BETWEEN fecha1 AND fecha2;
$$
language sql
```

# DECREMENTO DE STOCK

Decremento de cada stock basado en la venta. Si el producto está por debajo del estandar, lanzar mensaje, o si el producto está agotado no realizar venta.

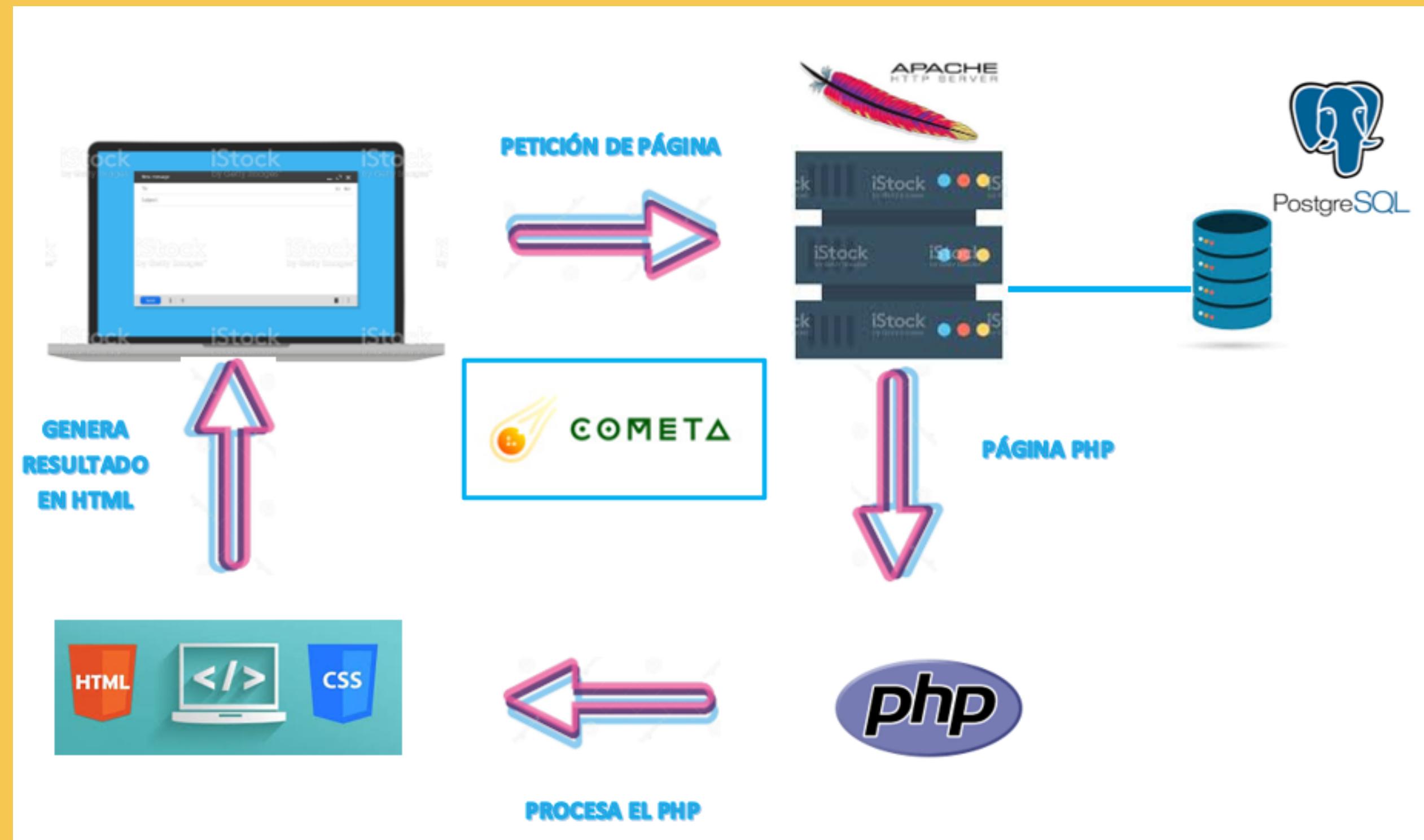
```
CREATE OR REPLACE PROCEDURE ingresa_Detalle (codigo_bar varchar(15), no_vent text, cantidad numeric)
LANGUAGE PLPGSQL
AS $$

DECLARE
    stockAct int;
    stockUp int;
BEGIN
    INSERT INTO DETALLE (codigo_barras, no_venta, cantidad)
    VALUES ($1, $2, $3);
    stockAct=(SELECT stock FROM PRODUCTO WHERE codigo_barras = $1);
    if(stockAct>$3) then
        UPDATE PRODUCTO SET stock= stockAct- $3
        WHERE codigo_Barras = $1;
        COMMIT;
    end if;
    stockUp = (SELECT stock FROM PRODUCTO WHERE codigo_Barras = $1);
    if(stockUp>0 and stockUp<3) then
        RAISE NOTICE 'El producto esta próximo a agotarse';
    elsif(stockUp= 0) then
        ROLLBACK;
    end if;
end;
$$;
```

# DIAGRAMA

## CONEXIÓN DE BASE DE DATOS Y PÁGINA WEB

Presentamos un diagrama donde muestra a grandes rasgos la conexión de base de datos y página web.



# Página Web

Muestra de página web y sus funciones.



# CONCLUSIÓN

Este proyecto ha sido un reto personal para ambas partes, pues al aplicar todos los conocimientos aprendidos de la materia de Bases de Datos, nos permitió conocer nuevas herramientas e incluso profundizar en algunas otras. Se presentaron retos, por lo cual tuvimos que recurrir a diversas fuentes tanto nacionales, como de otros idiomas, para despejar dudas y continuar con el diseño de la BD, así como con la codificación.

Ver los resultados de nuestro proyecto es satisfactorio, pues el trabajo en equipo siempre había sido un poco más fácil de manera presencial, sin embargo, trabajar de manera remota es más difícil, pues los tiempos deben coincidir, además existen otros problemas como el internet, ruido, tareas, etc., que sin duda alguna con este proyecto aprendimos a superar y llegar a acuerdos justo para ambas.



## DIRECCIÓN

Calle Rosa, 12, Toledo



## TELÉFONO

91-1234-567



## E-MAIL

frisse.corporation@frisse.com



DENISSE MARTÍNEZ RUIZ



denissemr@frisse.com



FRIDA E. MARTÍNEZ SILVA



fridaems@frisse.com

# DIRECTORIO