Segundo entregable del proyecto final

Entregable día 24 de Noviembre de 2022.-

El modelo de la base de datos corresponde a un comercio mayorista. Consta de una tabla correspondiente a el empleado. Sirve tanto para identificar al vendedor que realiza el pedido/venta, con todos los datos para identificarlo. Cuenta con una PK exclusiva de cada empleado para su identificación.

También cuenta con una tabla clientes, de la cual se obtienen los datos del cliente como su posteriores datos para el envió de la mercadería. También cuenta con una PK propia de cada Cliente.

Una tabla de proveedores, nuevamente con los datos de este, a fin de realizar compras al mismo y acreditar en el stock de la empresa. Esta tabla se relaciona con la tabla de Producto_stock, que es la que determina las existencias, nombre , descripción y precio final de venta de los productos que se comercializan.

Una tabla pedidos que consta de PK para cada orden de compra, se relaciona con las tablas Cliente y Empleado/Vendedor, a fin de determinar el vendedor y el comprador. Esta tambien se relaciona con la tabla del Producto_Stock de productos de la empresa que es donde se registra el Stock de la empresa. Tambien registra las cantidades y los precios de los productos que obtiene de la tabla Productos_Stock.

Como indique mas arriba la tabla de pedido se relaciona con la tabla de Producto_stock para cargar los productos del pedido en la misma. Esta registra las cantidades de stock y el proveedor de los mismos. Y los precios de compra de cada producto.

Y por ultimo tenemos la tabla de envíos a fin de tener control de los mismos, numero de rastreo, hasta la recepción del cliente.

Vistas del Entregable.
Esta vista muestra el id del cliente el nombre y el documento del comprador y lo une con la tabla pedidos con un inner join con pedidos que muestra el nombre del producto, las unidades ordenadas y la dirección del envió.

```
CREATE VIEW Cliente_pedido AS
SELECT C.id_cliente, C.nombre,C.Documento,P.Producto_nombre,
P.unidades_ordenadas, P.direccion_envio from cliente C
INNER JOIN pedidos P ON C.id_cliente = P.id_cliente;
```

Esta vista junta la tabla pedidos con los datos del empleado que la realizo

```
CREATE VIEW Empleado_venta_efectuada AS
SELECT P.orden_id, P.Producto_nombre, P.unidades_ordenadas, E.id_empleado,
E.nombre, E. Apellido FROM pedidos P
INNER JOIN empleado E ON P.id_empleado = E.id_empleado;
```

CREA una vista que une el id del cliente, la fecha de la orden, el numero de inventario del producto vendido, nombre del producto, unidades del mismo , direccion y del envio.

```
CREATE VIEW VENTAS AS
SELECT
P.orden_id,P.id_cliente,P.fecha_orden,P.sku_nro_producto_vendido,P.Producto_nombr
e,P.unidades_ordenadas,P.direccion_envio,S.precio_unidad  FROM pedidos P INNER
JOIN
producto_stock S ON P.sku_nro_producto_vendido = S.sku_nro_producto;
```

Crea una vista que une el id del envio, numero de seguimiento, empresa de transporte telefono y email de la misma con el id del cliente la fecha, el nombre del producto, la fecha de envio y la direccion.

```
CREATE VIEW envios AS
SELECT EN.id_envios, EN.numero_seguimiento, EN.empresa_de_transporte,
EN.telefono, EN.e_mail, PE.id_cliente, PE.fecha_orden,
PE.Producto_nombre,PE.fecha_envio, PE.direccion_envio FROM envio EN
INNER JOIN pedidos PE ON EN.orden_id = PE.orden_id;
```

```
CREATE VIEW Productos_Proveedor AS
SELECT PS.sku_nro_producto, PS.id_proveedor, PS.Producto_nombre, PS.Categoria,
PS.stock_unidades, PS.precio_unidad, PROV.Company_name, PROV.Cuit_proveedor,
PROV.telefono_proveedor,PROV.email_proveedor, PROV.home_page  FROM
producto_stock PS INNER JOIN proveedor PROV
ON PS.id_proveedor = PROV.id_proveedor ;
```

CARGA DE DATOS SOLO LA PRIMER ENTRADA(para que no sea muy largo igual esta el archivo en la carpeta).

1.-Tabla cliente :

```
insert into cliente (Nombre,Apellido, Documento, Cuit_Cliente, Pais, Provincia,
Direccion, Email, telefono)
values('Leandro', 'Passalacqua', '28488094', 2028, 'argentina', 'CABA', 'micasa 664
depto 2', 'leandro2311@gmail.com', '1566651102'),....
```

2.- Tabla empleado:

```
insert into empleado(nombre, apellido, Usuario, Clave_usuario, direccion,
fecha_de_ingreso, cargo, sucursal, telefono, e_mail_empleado, sueldo)
values('LEANDRO','PASSALACQUA','LEADIRECTOR','CLAVE123','Carlos Encina
664',23/11/2000,'Gerente','Casa Central','156651102','leandro@empresa.com', 200000),
```

3.-Tabla Proveedor:

```
insert into proveedor(Company_name,Cuit_proveedor, direccion_proveedor,
telefono_proveedor, email_proveedor, home_page)
```

values ('Apple s.a.',3012345,' One Apple Park Way; Cupertino, CA 95014, U.S.A.','8002752273','proveedores@apple.com','apple.com'),

4.-Table Producto:
INSERT INTO producto_stock (id_proveedor, Producto_nombre, Categoria, stock_unidades, precio_unidad)
values(1,'Iphone 14','smartphone',100,1000),

5.-Tabla pedidos
INSERT INTO pedidos (id_cliente, id_empleado, fecha_orden, sku_nro_producto_vendido,Producto_nombre,unidades_ordenadas, fecha_envio, direccion_envio)
values(1,2,'2022-11-23',6,'GeForce RTX 4080',1,'2022-11-23','mi casa 123'),

6.-Tabla Envios:
INSERT INTO envio
(orden_id,numero_seguimiento,empresa_de_transporte,telefono,e_mail)
values (1,111,'El Aguila', '333444', 'elaguila@transport.com')

TRIGGERS

-- CREA TABLA AUDITORIA, CREA TRIGGER AUDITORIA STOCK.

CREATE TABLE auditoria_stock (producto_nuevo_nombre VARCHAR (100),precio_nuevo INT, hora_creacion DATETIME);

CREATE TRIGGER trigger_auditoria_Stock AFTER INSERT ON producto_stock FOR EACH ROW INSERT INTO auditoria_stock
(producto_nuevo_nombre,precio_nuevo,hora_creacion)
VALUES (NEW.Producto_nombre,NEW.precio_unidad, NOW());

-- Insert para probar
INSERT INTO producto_stock (id_proveedor, Producto_nombre, Categoria, stock_unidades, precio_unidad) values (8,'productoX','ALGO',1,1);

SELECT * FROM producto_stock;

-- FUNCION 2 crea Una tabla de LOG de proveedores y le inserta la fecha de cuando fue creado el registro

CREATE TABLE proveedor_log (Nombre_proveedor VARCHAR (200), cuit_prov INT, fecha_registro DATETIME);

CREATE TRIGGER trigger_proveedor_ingresa AFTER INSERT ON proveedor FOR EACH ROW INSERT INTO proveedor_log (Nombre_proveedor,

```
cuit_prov,fecha_registro  ) VALUES (NEW.Company_name, NEW.Cuit_proveedor,
NOW());
-------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------
```
Stored Procedures

```
-- PROCEDIMIENTO PARA INSERTAR Y MOSTRAR DATOS EN TABLA


DROP PROCEDURE IF EXISTS inserta_producto;
DELIMITER //
CREATE PROCEDURE inserta_producto ( IN sp_id_proveedor INT, IN
sp_Producto_nombre varchar(200), IN sp_Categoria  varchar(100), IN
sp_stock_unidades INT, IN sp_precio_unidad INT )
BEGIN
        INSERT INTO producto_stock
(id_proveedor,Producto_nombre,Categoria,stock_unidades,precio_unidad)
   VALUES
(sp_id_proveedor,sp_Producto_nombre,sp_Categoria,sp_stock_unidades,sp_precio_uni
dad);
        SELECT * FROM producto_stock;
END //

-----------------------------------------
-- PROCEDIMIENTO PARA ORDENAR POR HEADER
-- Este Procedimiento ordena de manera descendente y ascendenmte, cuando es llamada
y dado el parametro del campo a ordenar que en este caso son `orden_id` int NOT
NULL AUTO_INCREMENT,
--  `id_cliente` ,`id_empleado` ,  `fecha_orden`,  `sku_nro_producto_vendido`,
`Producto_nombre` ,`unidades_ordenadas`, `fecha_envio` ,  `direccion_envio`
,`precio_venta`
-- y un segundo parametro ASC para Ascendente o DESC para descendente

DROP PROCEDURE IF EXISTS sp_mi_procedimiento;
DELIMITER //
   CREATE PROCEDURE sp_mi_procedimiento(IN campo_orden varchar(50), IN
tipo_orden varchar(5))
      BEGIN
         SET @s = CONCAT('SELECT * FROM pedidos ORDER BY ',campo_orden,'
',tipo_orden);
         PREPARE stmt2 FROM @s;
         EXECUTE stmt2;
   DEALLOCATE PREPARE stmt2;
      END //
DELIMITER //

-------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------
```

Funciones:

```
--  FUNCION PARA EL AÑO BISIESTO DETERMINA SI EL AÑO ES BISIESTO O NO
DELIMITER //
CREATE FUNCTION  bisiesto (fecha INT) RETURNS varchar(25)
DETERMINISTIC
BEGIN
DECLARE a VARCHAR(20);
IF fecha % 4 = 0 AND (fecha % 100 !=0 OR fecha % 400=0) THEN
RETURN 'Es bisiesto';
ELSE
RETURN 'NO es bisiesto';
END IF;
END//
DELIMITER ;

-- FUNCION CALCULA BENFICIO

DELIMITER //
CREATE FUNCTION calcularGanancia(coste INT, precio INT) RETURNS INT
DETERMINISTIC
BEGIN
        DECLARE beneficio INT;
        SET beneficio = precio - coste;
        RETURN beneficio;
END //
DELIMITER ;
```

En la carpeta estan los arrchivos .sql del resto de las entregas.