****

**Tecnicatura Universitaria en Programación**

**Laboratorio de Computación III**

**Equipo # 01**

**Integrantes:**

**Zurita, Maximiliano - 24291**

**Gianfranco, Pantanetti - 28949**

**Fernandez, Ramiro – 28819**

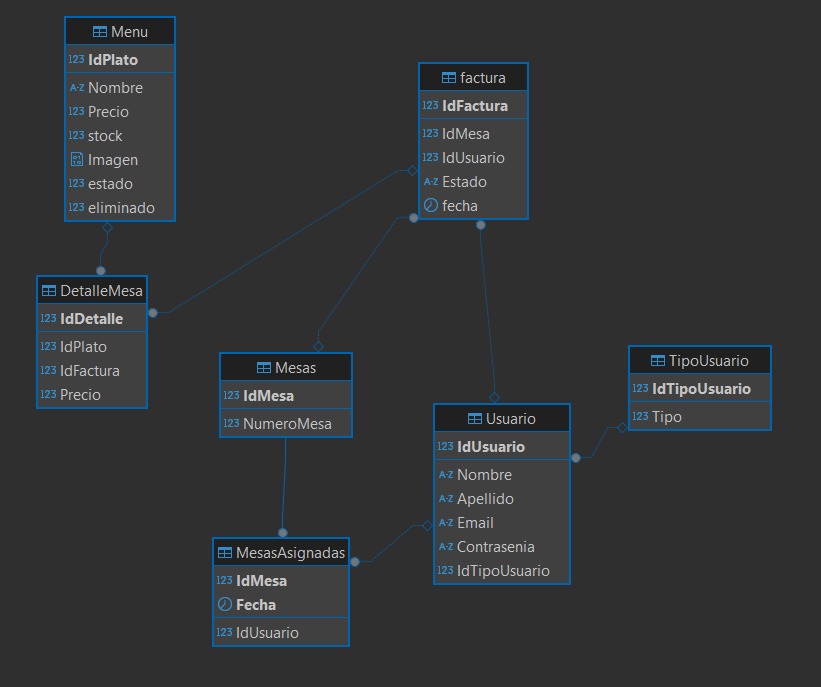
**Explicación del Sistema**

El sistema desarrollado permite gestionar un restaurante de forma eficiente, abarcando desde la administración de usuarios (mozos, clientes, administradores), la gestión de mesas y menús, hasta la facturación y asignación de mozos.  
  
Incluye control de permisos por tipo de usuario, generación de facturas detalladas, y un registro de las órdenes realizadas en cada mesa. Se implementan distintas funcionalidades orientadas a facilitar las tareas operativas del personal y mejorar la atención al cliente.

**Funcionalidades principales**

* **Gestión de usuarios:** Administración de distintos tipos de usuario (administrador, mozo, cliente), con sus datos personales y credenciales.
* **Gestión de mesas:** Alta, baja y modificación de las mesas del restaurante, permitiendo su asignación a clientes y mozos.
* **Gestión de menús:** Administración de productos disponibles en el restaurante, incluyendo su categoría, precio y estado.
* **Toma de pedidos y facturación:** Registro de ventas asociadas a mesas, mozos y clientes, con cálculo automático del total.
* **Historial de ventas:** Consulta detallada de ventas realizadas, con opción de filtrar por usuario, mozo o fecha.

**Diagrama de Entidad-Relación**



**Objetos de Base de Datos clave en el sistema**

**Vista 1: vw\_MesasAsignadasHoy**

SELECT

m.IdMesa,

m.NumeroMesa,

u.Nombre + ' ' + u.Apellido AS NombreMozo,

f.IdFactura,

f.Estado AS EstadoFactura

FROM

MesasAsignadas ma

JOIN Mesas m ON ma.IdMesa = m.IdMesa

LEFT JOIN Usuario u ON ma.IdUsuario = u.IdUsuario

LEFT JOIN factura f ON f.IdMesa = m.IdMesa AND f.Estado = 'ABIERTA'

WHERE

ma.Fecha = CAST(GETDATE() AS DATE);

**Vista 2: vw\_MesasConMasFacturacion**

Devuelve un listado con los productos más vendidos del sistema, ordenados por cantidad.

SELECT

m.IdMesa,

COUNT(v.ID) AS CantidadVentas,

SUM(v.Suma\_total) AS TotalFacturado

FROM Ventas v

JOIN Mesa m ON v.IdMesa = m.IdMesa

GROUP BY m.IdMesa

ORDER BY TotalFacturado DESC;

go

CREATE VIEW VW\_PedidosPorDiaYHora AS

SELECT

CONVERT(date, v.Fecha\_venta) AS Fecha,

DATEPART(HOUR, v.Fecha\_venta) AS Hora,

COUNT(\*) AS CantidadPedidos

FROM Ventas v

GROUP BY CONVERT(date, v.Fecha\_venta), DATEPART(HOUR, v.Fecha\_venta)

ORDER BY Fecha, Hora;

**Procedimiento Almacenado 1: sp\_AsignarMesero**

create or alter procedure sp\_AsignarMesero(

@NumeroMesa int,

@IdMesero int

)as

begin

declare @IdMesa int;

select @IdMesa = IdMesa from mesas where NumeroMesa = @NumeroMesa

IF NOT EXISTS (SELECT \* FROM MesasAsignadas WHERE IdMesa = @IdMesa AND fecha = CAST(GETDATE() AS DATE))

BEGIN

INSERT INTO MesasAsignadas (IdMesa, IdUsuario, Fecha)

VALUES (@IdMesa, @IdMesero, GETDATE())

END

ELSE

BEGIN

UPDATE MesasAsignadas

SET IdUsuario = @IdMesero

WHERE IdMesa = @IdMesa

END

end

**Procedimiento Almacenado 2: sp\_AsignarPlato**

create or alter procedure sp\_AsignarPlato(

@IdPlato int,

@IdMesa int

)as

begin

DECLARE @IdFactura int

declare @Stock int

declare @Precio money

select @IdFactura = IdFactura from factura f where @IdMesa = IdMesa and f.Estado = 'ABIERTA'

select @Precio = m.Precio from Menu m

WHERE m.IdPlato = @IdPlato

insert into DetalleMesa (IdPlato,IdFactura,Precio) values (@IdPlato,@IdFactura,@Precio)

update Menu

set stock = stock - 1

where IdPlato = @IdPlato

select @Stock = stock from menu

where IdPlato = @IdPlato

if @Stock <= 0

begin

update Menu

set Estado = 0, stock = 0

where IdPlato = @IdPlato

end

end

**Trigger 1: TR\_ValidarStockAlVender**

CREATE TRIGGER TR\_ValidarStockAlVender

ON VentaProducto

INSTEAD OF INSERT

AS

BEGIN

IF EXISTS (

SELECT 1

FROM INSERTED i

JOIN Producto p ON i.IdProducto = p.IdProducto

WHERE i.Cantidad > p.Stock

)

BEGIN

RAISERROR('No hay stock suficiente para uno o más productos.', 16, 1);

ROLLBACK;

END

ELSE

BEGIN

-- Insertar venta y actualizar stock

INSERT INTO VentaProducto (IdVenta, IdProducto, Cantidad)

SELECT IdVenta, IdProducto, Cantidad

FROM INSERTED;

UPDATE p

SET p.Stock = p.Stock - i.Cantidad

FROM Producto p

JOIN INSERTED i ON p.IdProducto = i.IdProducto;

END

END;

**Trigger 2:**

**Links a los recursos**

**Script de creación de base de datos con datos**

<https://github.com/maximilianozurita/tp-labo3>

**Video demo del sistema**

[https:/](https://youtube.com/usuario/VideoDemo)