****

**Tecnicatura Universitaria en Programación**

**Laboratorio de Computación III**

**Equipo # 01**

**Integrantes:**

**Zurita, Maximiliano - 24291**

**Gianfranco, Pantanetti - 28949**

**Fernandez, Ramiro – 28819**

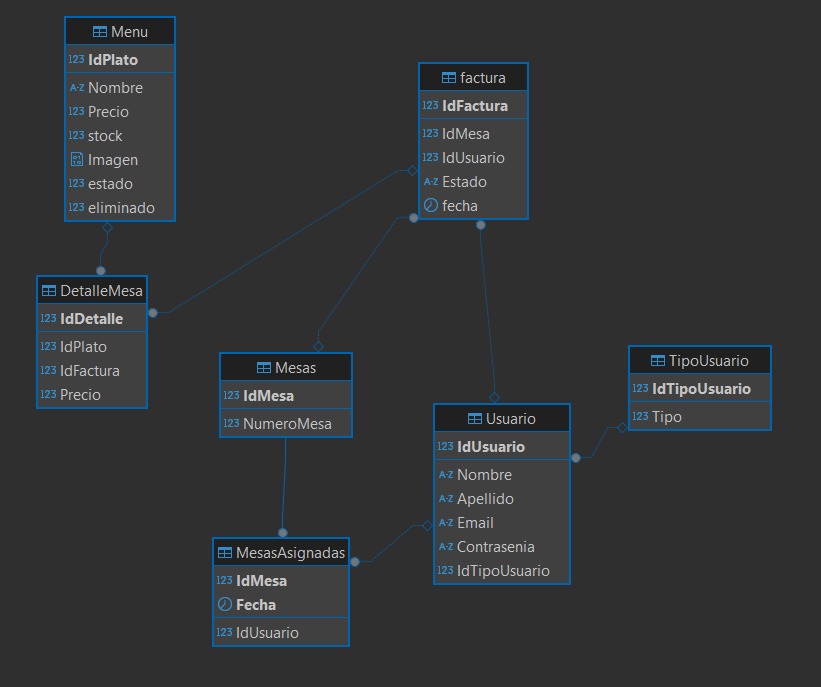
**Explicación del Sistema**

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**

CREATE VIEW vw\_MesasAsignadasHoy AS

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 Usuarios u ON ma.IdUsuario = u.IdUsuario

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

WHERE

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

go

**Vista 2: vw\_FacturacionDiariaDetallada**

CREATE VIEW vw\_FacturacionDiariaDetallada AS

SELECT

f.IdFactura,

f.IdMesa,

m.NumeroMesa,

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

me.Nombre AS Plato,

dm.Precio,

f.Fecha

FROM

Facturas f

JOIN Usuarios u ON f.IdUsuario = u.IdUsuario

JOIN DetalleMesas dm ON f.IdFactura = dm.IdFactura

JOIN ItemsDelMenu me ON dm.IdPlato = me.IdPlato

JOIN Mesas m ON f.IdMesa = m.IdMesa

WHERE

f.Fecha = CAST(GETDATE() AS DATE) AND f.Estado = 'CERRADO';

go

**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

GO

**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 Facturas f where @IdMesa = IdMesa and f.Estado = 'ABIERTA'

select @Precio = m.Precio from ItemsDelMenu m

WHERE m.IdPlato = @IdPlato

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

update ItemsDelMenu

set stock = stock - 1

where IdPlato = @IdPlato

select @Stock = stock from ItemsDelMenu

where IdPlato = @IdPlato

if @Stock <= 0

begin

update ItemsDelMenu

set Estado = 0, stock = 0

where IdPlato = @IdPlato

end

end

go

**Trigger 1: TR\_ValidarStockAlVender**

CREATE TRIGGER TR\_ValidarStockAlVender

ON DetalleMesas

INSTEAD OF INSERT

AS

BEGIN

IF EXISTS (

SELECT 1

FROM INSERTED i

JOIN ItemsDelMenu p ON i.IdPlato = p.IdPlato

WHERE i.Precio > 0 AND p.Stock < 1

)

BEGIN

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

RETURN;

END;

INSERT INTO DetalleMesas (IdPlato, IdFactura, Precio)

SELECT IdPlato, IdFactura, Precio

FROM INSERTED;

UPDATE p

SET p.Stock = p.Stock - 1

FROM ItemsDelMenu p

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

END;

GO

**Trigger 2:**

CREATE TRIGGER trg\_InsertVentaOnFacturaCerrada

ON Facturas

AFTER UPDATE

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO Ventas (IdFactura, IdMesa, FechaVenta, SumaTotal)

SELECT

f.IdFactura,

f.IdMesa,

f.Fecha,

ISNULL(SUM(dm.Precio), 0)

FROM

inserted i

JOIN Facturas f ON i.IdFactura = f.IdFactura

LEFT JOIN DetalleMesas dm ON dm.IdFactura = f.IdFactura

JOIN deleted d ON d.IdFactura = i.IdFactura

WHERE

i.Estado = 'CERRADO' AND d.Estado <> 'CERRADO'

AND NOT EXISTS (

SELECT 1 FROM Ventas v WHERE v.IdFactura = i.IdFactura

)

GROUP BY f.IdFactura, f.IdMesa, f.Fecha;

END;

GO

**Links a los recursos**

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

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

**Script Storage procedures:**

<https://github.com/maximilianozurita/tp-labo3/tree/main/queries/Storage%20Procedures>

**Script Views:**

<https://github.com/maximilianozurita/tp-labo3/tree/main/queries/Views>

**Script Triggers:**

<https://github.com/maximilianozurita/tp-labo3/tree/main/queries/Views>

**Video demo del sistema**

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