****

**Tecnicatura Universitaria en Programación**

**Laboratorio de Computación III**

**Equipo # 01**

**Integrantes:**

**Zurita, Maximiliano - 24291**

**Gianfranco, Pantanetti - 28949**

**Fernandez, Ramiro – 28819**

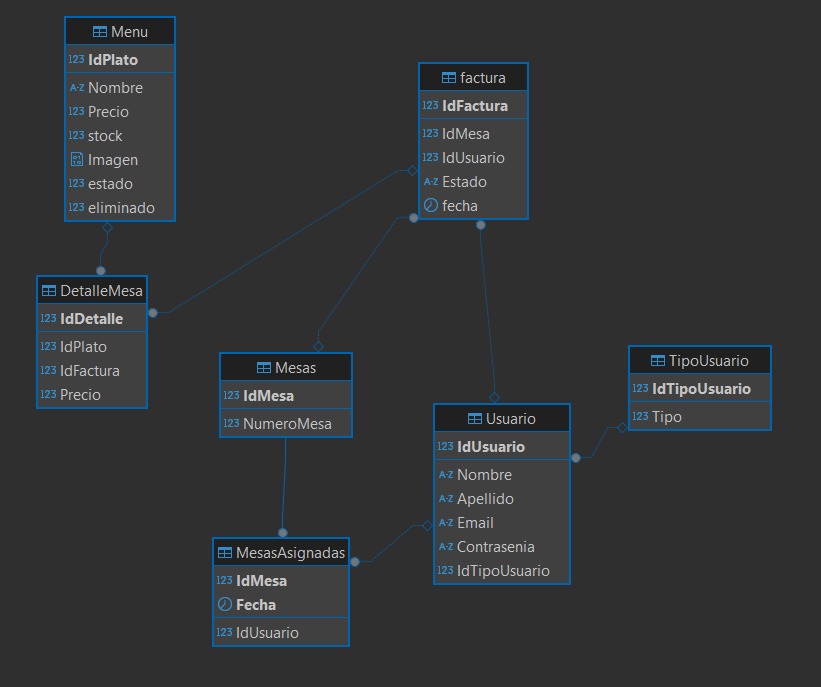
**Explicación del Sistema**

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**

Esta vista muestra las mesas que fueron asignadas en el día actual, incluyendo el número de mesa, el nombre del mozo asignado y la factura abierta asociada, si la hay. Se filtra por la fecha actual (GETDATE()) para limitar los resultados solo a las asignaciones del día. Usa LEFT JOIN para incluir tanto al mozo como a la factura, incluso si no existen.

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**

Muestra el detalle de todas las facturas cerradas en el día, incluyendo los platos vendidos, sus precios, la mesa asociada y el mozo que atendió. Esto permite conocer exactamente qué se vendió y a qué precio durante la jornada. La vista se enfoca solo en facturas con estado 'CERRADO' y fecha del día.

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

**Vista 3: vw\_** **MesasConMasFacturacion**

Agrupa todas las ventas por mesa y calcula cuántas ventas tuvo cada una (CantidadVentas) y el total facturado (TotalFacturado). Es útil para detectar qué mesas generan más ingresos y cuán utilizadas están. Se apoya en la tabla Ventas y se asocia con Mesas para mostrar el ID correspondiente.

CREATE VIEW VW\_MesasConMasFacturacion AS

SELECT

m.IdMesa,

COUNT(v.IdMesa) AS CantidadVentas,

SUM(v.SumaTotal) AS TotalFacturado

FROM Ventas v

JOIN Mesas m ON v.IdMesa = m.IdMesa

GROUP BY m.IdMesa

go

**Vista 4: vw\_** **PedidosPorDiaYHora**

Esta vista agrupa los pedidos realizados por día y hora, mostrando cuántos pedidos se realizaron en cada franja horaria. Es útil para análisis de demanda por horario, permitiendo al negocio identificar los momentos de mayor actividad. Se usa DATEPART(HOUR, ...) para extraer la hora de cada venta.

CREATE VIEW VW\_PedidosPorDiaYHora AS

SELECT

CONVERT(date, v.FechaVenta) AS Fecha,

DATEPART(HOUR, v.FechaVenta) AS Hora,

COUNT(\*) AS CantidadPedidos

FROM Ventas v

GROUP BY CONVERT(date, v.FechaVenta), DATEPART(HOUR, v.FechaVenta)

go

**Procedimiento Almacenado 1: sp\_AsignarMesero**

Este procedimiento asigna un mozo (mesero) a una mesa específica. Si la mesa aún no fue asignada en el día actual, se inserta una nueva entrada en la tabla MesasAsignadas. En caso contrario, se actualiza el mozo asignado a la mesa. La fecha se compara contra el día actual (GETDATE() casteado a DATE) para asegurar que la asignación sea por jornada.

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**

Este procedimiento registra un plato pedido por una mesa, asociándolo a su factura abierta. Se agrega el plato a la tabla DetalleMesas con su precio correspondiente, y se descuenta una unidad del stock del producto. Si el stock alcanza 0 o menos, el plato se marca automáticamente como inhabilitado (Estado = 0) y su stock se ajusta a cero para evitar futuras ventas.

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\_PrevenirFacturasDuplicadasAbiertas**

Este trigger sirve para evitar que se cree más de una factura abierta para la misma mesa. Se activa cuando se intenta insertar una nueva factura. La tabla virtual inserted contiene los datos que se quieren insertar la cual se compara con la tabla Facturas para ver si ya existe una factura con la misma mesa (IdMesa) y con estado 'ABIERTA'. Si ya existe una factura abierta para esa mesa, se considera un conflicto.

Esto protege la integridad de los datos, impidiendo tener más de una factura activa por mesa al mismo tiempo.

CREATE OR ALTER TRIGGER TR\_PrevenirFacturasDuplicadasAbiertas

ON Facturas

INSTEAD OF INSERT

AS

BEGIN

SET NOCOUNT ON;

IF EXISTS (

SELECT 1

FROM inserted i

JOIN Facturas f ON i.IdMesa = f.IdMesa

WHERE i.Estado = 'ABIERTA' AND f.Estado = 'ABIERTA'

)

BEGIN

RAISERROR('Ya existe una factura ABIERTA para esta mesa.', 16, 1);

RETURN;

END

INSERT INTO Facturas (IdMesa, IdUsuario, Estado, Fecha)

SELECT IdMesa, IdUsuario, Estado, Fecha

FROM inserted;

END;

GO

**Trigger 2 trg\_InsertVentaOnFacturaCerrada:**

Este trigger se dispara después de actualizar la tabla Facturas, y registra automáticamente una venta en la tabla Ventas cuando una factura cambia su estado a 'CERRADO'. Calcula el total facturado sumando los precios de los ítems asociados y evita duplicados mediante una verificación con la tabla Ventas. Esto automatiza el cierre de ventas y mantiene la trazabilidad de las operaciones.

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/Trigger>

**Video demo del sistema**

<https://drive.google.com/file/d/1URb8Gt9dj12Xz-POVbPbEPKADiIZsaFo/view?usp=drive_link>